AUGUST 1984 ÖS 50,–/Sfr 6,– DM 6,–




8|84

# DAS MAGAZIN FÜR COMPUTER-FANS

Tuning durch andere Sprachen

## Ist Basic out?

## Große Übersicht: Modems/Akustikkoppler

Keine Probleme mit der Floppy

## Selbsthilfe: VC 1541-Tricks

Vergleichstest: EPROM-Brenner

## Programm auf Knopfdruck

Listing des Monats: Das außergewöhnliche Abenteuerspiel

## Burg des Grauens

## So nutzt man den Videochip des VC 20 voll aus

Softwaretest: ISM 64

## Leistungsfähige Datenbanken schnell erstellt

Listings: Hardcopy für Görlitz-Interface ★ Grafik in Texte einblenden ★ Programme zum Messen, Steuern, Regeln ★ Reset-Taste ★ Test: Paint Magic ★ Kurse ★ Viele Tips und Tricks für den VC 20 und Commodore 64

Eine Publikation der Markt & Technik Verlag Aktiengesellschaft

## INHALT

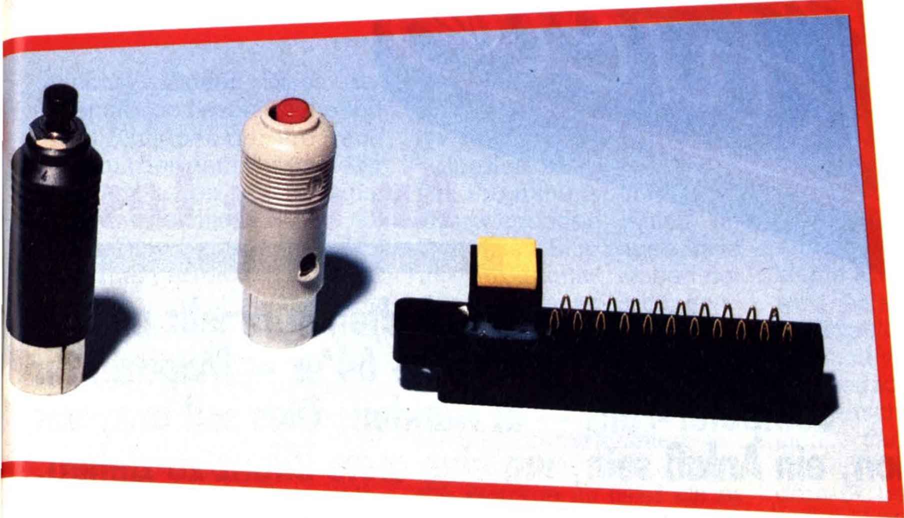




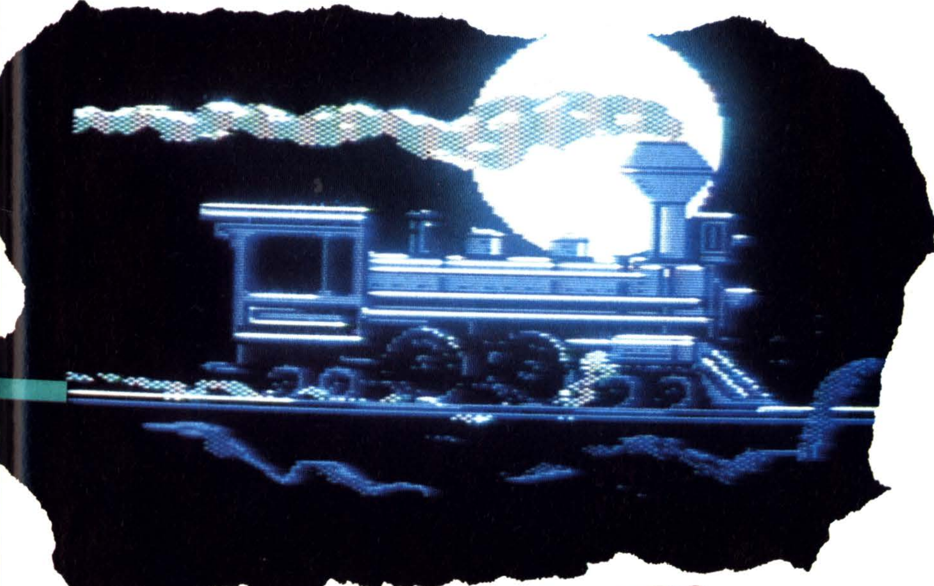

◄ Auf der CES in Chicago gab es jede Menge neuer Produkte zu bewundern. Was wird uns in der Zukunft erwarten?

► Modems und Akustikkoppler sind ein heißes Thema. Wer bietet was für seinen Preis? Wir stellen verschiedene Modems und Akustikkoppler vor.

Die Geheimnisse der Floppy lüften. Wir geben Reparaturhinweise. ▲



▲ Resettaster und -Schalter in allen Formen und Variationen. Doch wie geht's danach weiter?



▲ Mit Paint Magic lassen sich nahezu problemlos Bilder auf dem C 64 erstellen.



▲ Der Commodore 64 im Kuhstall. Eine nicht ganz abwegige Anwendung.



## Software-Test



## Spiele-Test



## Kurse



## So machen's andere



## Anwendung des Monats



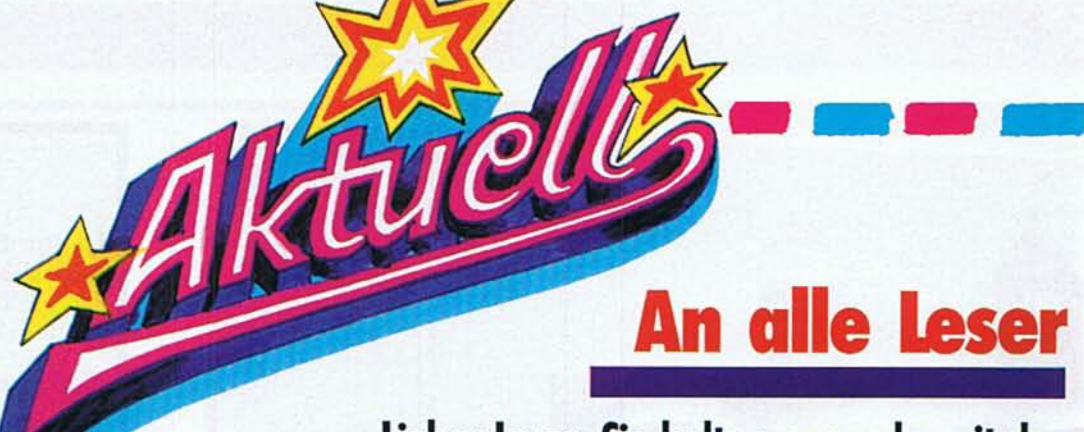
## Listing des Monats



## Wettbewerbe



## Rubriken




# An alle Leser

**Lieber Leser, Sie halten nunmehr mit der Ausgabe 8/84 das fünfte 64'er — Magazin für Computer-Fans — in Händen. Dies soll uns, der Redaktion, ein Anlaß sein, um eine erste Bilanz zu ziehen.**

Der Commodore 64 hat im Verbund mit seinem Vorgänger dem VC 20 mittlerweile einen Marktanteil von über 50 Prozent bei den Heimcomputern erreicht. Diese Entwicklung war Anfang dieses Jahres in der Tendenz bereits abzusehen. Die entsprechende Reaktion auf diese Entwicklung war die Absicht, eine eigene Zeitschrift für diese beiden Computer zu machen. Nach dreimonatigen konzeptionellen Arbeiten mit nur zwei aktiven Redakteuren war die 64'er, Ausgabe 4/84 fertig.

Unser Team wurde während dieser Zeit um zwei weitere, versierte Fachleute verstärkt. Der momentan stattfindende Arbeitskampf der Druck- und Metallindustrie um die 35-Stunden-Woche läßt uns nur müde lächeln. Wir kämpfen um die 70-Stunden-Woche. Unser momentanes Arbeitspensum beträgt 75 bis 80 Stunden pro Woche. Sie werden sich fragen, warum dieser Arbeitsaufwand? Einige Zahlen sollen dies veranschaulichen. Wir bekommen pro Woche etwa 100 Programmeinsendungen und ungefähr die Hälfte an Briefzuschriften pro Tag. Dabei sind die persönlichen Besuche von Lesern und Programmentwicklern in der Redaktion nicht mitgerechnet. Die Bearbeitung der Programme, die Beantwortung der Briefe und der persönlichen Probleme per Telefon nimmt entsprechend Zeit in Anspruch. Auch unsere beiden Schwesterzeitschriften Computer persönlich und Happy-Computer wollen von uns mit Commodore-Programmen und -Artikeln versorgt werden. »Es gibt viel zu tun, ...«.

Warum nun dieser Brief an die Leser? Wir wollen uns nicht beschweren über die unheimliche Resonanz, die unserer 64'er-Magazin hervorgerufen hat. Im Gegenteil, wir sind stolz darauf, zeigt es uns doch, daß wir auf dem richtigen Weg sind. Dennoch sollen im folgenden einige Punkte aufgeführt werden, die der Redaktion die Arbeit und dem Leser die Mitarbeit an unserer Zeitschrift erleichtern sollen.

## Neun kleine Negerlein...

1. Die Mitmach-Karten im 64'er haben sich als das erwiesen was sie sein sollten: ein Forum für die Leser, sich ihre Probleme von der Seele zu schreiben. Für uns sind sie ein wertvoller Hinweis dafür, wo Sie der Schuh drückt. Die ausgefüllte Rubrik »Ich wünsche mir für die nächsten Hefte folgende Themen:« ist ein wichtiger Beitrag, was wir in den zukünftigen Ausgaben unbedingt berücksichtigen müssen. So wird zum Beispiel das am meisten gewünschte Thema, ein Maschinensprach-Kursus, ab der nächsten Ausgabe in Angriff genommen. Ein Musikkurs wird ebenfalls bald folgen.

Mit dem Teil »Ich stehe vor folgendem Problem:« können Sie Fragen an die Redaktion stellen. Es ist uns dabei aber nicht möglich, alle Fragen zu beantworten. Zum einen handelt es sich oft um sehr ausgefallene Themen, wie zum Beispiel der Anschluß eines »Exoten-Druckers« an den C 64. Zum anderen werden aber auch viele Fragen durch Artikel oder Programme, die wir bereits veröffentlicht haben oder noch werden in der einen oder anderen Form geklärt.

»In dieser Ausgabe war besonders gut« und »Deshalb meine Meinung ..« werden zur Analyse der erschienen Ausgaben verwendet. Diese Analyse liefert wertvolle Hinweise, was gefallen hat oder nicht, oder wo noch Verbesserungen anzubringen sind.

2. Wir haben mittlerweile vielfältige Möglichkeiten Ihr Listing zu verwerten. Zum einen natürlich die Veröffentlichung in unserer Zeitschrift. Bei sehr guten Programmen kann die Markt & Technik Verlag Aktiengesellschaft das Produkt sogar in den Vertrieb von Happy Software aufnehmen. Ferner besteht die Möglichkeit, daß Ihr Listing als Teil eines Buches veröffentlicht werden kann.

3. Uns erreichen auch immer wieder Anfragen, ob und in welcher Form abgedruckte Programme auf einem Datenträger zu erstehen sind. Wir haben bereits in der Ausgabe 7/84 (zunächst für die Ausgaben 4 und 5) begonnen alle im 64'er abgedruckten Programme auf Kassette anzubieten. Der Verkauf auf Diskette wird, zum Selbstkostenpreis versteht sich, dem-



## Piraten, Piraten

Softwarepiraten nannte man bislang Leute, die fremde Programme ohne Genehmigung kopieren und weiterverkaufen. Die Inhaber einer Scheinfirma namens R + S (Adresse: Ein Postfach in Berlin) haben eine neue Art von Softwarepiraterie erfunden: Sie verschickten serienweise Abmahnbriefe an Inserenten, die in Kleinanzeigen Software zum Kauf oder Tausch anboten; behaupteten, daß es sich dabei um Raubkopien handle und verlangten Unterzeichnung einer Unterlassungserklärung sowie Begleichung einer Rechnung für die ihre angeblichen Kosten von 300 Mark — in bar an die Postfachadresse zu schicken. Sie hofften, ein Geschäft mit der Unwissenheit oder dem schlechten Gewissen der Adressaten zu machen. Keine Sorge: Um diese »Firma« kümmert sich schon der Staatsanwalt.

Die Reaktion auf diese Briefe zeigten aber häufig, wie unsicher die Empfänger waren, was sie eigentlich dürfen und nicht dürfen. Dabei ist das Grundprinzip seit langem klar: Ein Programm, das ein anderer geschrieben hat, darf man nur mit — zweckmäßigerweise schriftlicher — Genehmigung des Autors kopieren und Programme, die von vornherein zum Kopieren bestimmt sind, gibt es zwar — sie sind aber nicht gerade häufig. Auch solche Programme dürfen in der Regel nicht kommerziell weiterverwertet (in der Praxis heißt das zumeist: nicht weiterverkauft) werden. Sie dürfen auch nicht das 64er oder eine andere Zeitschrift oder ein Buch auf den Fotokopierer legen und die Kopien regelmäßig im Benutzerclub verteilen oder einen Datenträger mit den abgetippten Programmen weiterverkaufen. So einfach ist das.

Michael Pauly, Chefredakteur

nächst anlaufen. Dieser regelmäßige Leserservice soll das Eintippen der Programme mit den damit verbunden Fehlerquellen für Interessierte erleichtern.

4. Eine Unmenge an Zuschriften erreicht uns zu angeblich nicht lauffähigen Programmen. Alle Listings, die im 64'er abgedruckt worden sind, wurden von uns geprüft und für gut befunden. Dennoch kann es von Fall zu Fall vorkommen, daß sich Übertragungsfehler einschleichen. Bestes Beispiel ist der nicht gedruckte »Linkspfeil« im Listing von Quicktext. Sollten Sie in der Rubrik »Fehlerteufelchen« keine Korrektur zu einem Programm finden, so ist das Listing 100 prozentig lauffähig. Wie die Erfahrung gezeigt hat, lassen sich fast alle Probleme auf Tippfehler oder Nichtbeachtung der Anleitung reduzieren. Wir drucken nur ausgesuchte Programme ab. Diese enthalten dann fast immer eine sehr große Anzahl von DATA-Zeilen. Gerade hier ist natürlich die Wahrscheinlichkeit eines falsch eingegebenen Datums sehr hoch. Auch nach einer drei- fünf- oder zehnmaligen Überprüfung überliest man seine eigenen Fehler immer wieder. Bitte schreiben oder rufen Sie nur an, wenn Sie absolut sicher sind, daß in dem Programm Fehler sind.

Es tauchen aber oft bereits Schwierigkeiten beim Eintippen der Programme auf. So stehen manche Anfänger ratlos vor Zeilen mit mehr als 80 Zeichen oder können dies und jenes Steuerzeichen nicht eingeben. Wir können dem Leser leider nicht das intensive Studium des Handbuchs (zum Beispiel Anhang D) abnehmen. Des weiteren sollte natürlich auch die Anleitung zu dem Programm aufmerksam gelesen werden. Ein Listing einzutippen das nur unter Simons Basic läuft, ohne aber diese Spracherweiterung zu besitzen kann nicht zum Erfolg führen.

Beachten Sie in dieser Hinsicht auch unsere Tips zur Fehlersuche in Programmen (64'er, 7/84, Seite 66).

Speziell an die Einsender von Programmen mit einem überwiegenden Anteil an Datazeilen. Bitte bauen Sie in Ihr Programm unbedingt Checksummen-Prüfungen ein. Diese sollten sich nicht nur auf eine Gesamt-Checksumme beschränken, sondern den möglichen Fehler lokal eingrenzen helfen.

5. Manche Leser finden die Beiträge im 64'er als zu speziell, zu fachorientiert und nur dem Profi zugänglich, defacto für den Laien nicht verwertbar. Die Profis betrachten wiederum andere Artikel als zu wenig in die Tiefe gehend und informativ für ihre Zwecke. Dieser Problematik bei der Auswahl der Themen sind wir uns durchaus bewußt. Wir wollen aber ein Magazin sein, das sowohl dem Anfänger als auch dem Fortgeschrittenem einiges bietet. Dadurch ergibt sich eine Art Zweiteilung unserer Beiträge. Sowohl der Einsteiger in die Computerei (mit dem C 64 und VC 20) als auch der Könner sollen durch das 64'er Hilfestellung erfahren. Gerade bei unseren Kursen zeigt sich dies besonders deutlich. Hier ist wirklich für jeden etwas dabei.

6. Einige Leser beschweren sich auch, daß in einem Magazin mit dem Namen 64'er — zumindest zuviele — VC 20-Programme abgedruckt werden, andere sind gerade der gegenteiligen Auffassung. Der VC 20 ist der kleine Bruder vom C 64 und wird, solange er in großen Stückzahlen auf dem Markt ist, auch von uns mit Beiträgen bedacht. Der Titel unserer Zeitschrift bedeutet also nicht, daß wir uns ausschließlich mit dem Commodore 64 befassen. Dadurch lassen sich auch die zukünftigen Heimcomputer von Commodore einbinden.

7. Gefordert wird auch oft die Umschreibung der Programme für den VC 20 oder den C 64. Dies haben wir bisher wegen des erheblichen Platzbedarfs für ein und dasselbe Listing vermieden. Leser, die Programme selbst umschreiben möchten, seien auf den Artikel in der 64'er, Ausgabe 7, Seite 52 verwiesen.

8. Viele haben gefragt, warum wir bei den Software-Tests keine festen Bewertungen mit Nummern oder Buchstaben einführen. Dazu folgendes; jedes Produkt, das von uns getestet wird, ist es auch wert. Nur gute Hard- oder Software wird von uns besprochen und erhält entsprechenden Raum im 64'er. Der Markt wird monentan von Produkten überflutet, und wir wollen nur das Beste vorstellen. Vergleichende Bewertungen mit abschließenden Zahlen sollen der Objektivität dienen, sind allerdings zu oft das Ergebnis einer subjektiven Beurteilung. Deshalb verzichten wir darauf, liefern Fakten und persönliche Meinungen, überlassen es aber ansonsten dem mündigen Leser sich ein Urteil zu bilden, welches Produkt nun für seinen speziellen Anwendungsfall am besten geeignet ist.

9. Für alle Leser die erst jetzt auf das 64'er gestoßen sind, gibt es eine traurige Nachricht: alle Ausgaben 4 und 5 sind vergriffen. Nachbestellungen haben also keinen Zweck mehr. Artikel oder Programme die Sie interessieren müssen Sie sich also bei Freunden oder Bekannten besorgen, die diese »Raritäten-64'er« besitzen. Es ist uns leider nicht möglich einzelne Beiträge als Kopie zu verschicken.

Diese Ausführungen sollen Sie nicht abschrecken, weiterhin Programme, Artikel, Fragen und Meinungen an uns zu senden. Wir legen sehr großen Wert auf einen positiven und konstruktiven Dialog mit unseren Lesern. Um Verständnis wäre noch zu bitten, bei den doch recht langen Bearbeitungszeiten. Wir arbeiten daran — 12 Stunden am Tag, die ganze Woche durch. (aa)

## Ein neues Gesicht

Viele Leser haben oft zu Recht unsere Art und Weise, die Listings abzudrucken kritisiert. Da wurden Listings schräg gestellt, so daß man sich beim Abtippen den Kopf verdrehte, oder der Text mit einer Farbe unterlegt die das Lesen besonders unter Kunstlicht erschwerte. Wir haben die Anregungen der Leser aufgenommen und werden ab dieser Ausgabe mit einem relativ neuem Gesicht bei den Listings aufwarten.

Wo liegen nun die Verbesserungen? Zum einen werden die Listings in Zukunft mit einer einheitlichen Spaltenbreite (nämlich 40 Zeichen pro Zeile) gedruckt. Dadurch ist ein sofortiger Vergleich des Druckerlistings mit der Bildschirmdarstellung (zumindest beim C 64) gegeben. Zum anderen findet der Abdruck nahezu im Maßstab 1:1 statt, also kein Ärger mehr mit zu kleinen Listings.

Auch die Farbe ist weg, nur noch schwarz auf weiß.

Wenn Sie die neue Gestaltung von Seite 69 bis Seite 111 betrachten, so werden Sie feststellen, daß auch der Text in einem veränderten Format erscheint. Gerade in den Beschreibungen zu den Listings kommen immer wieder Basic-Zeilen vor, diese lassen sich dann wesentlich leichter lesen.

Wir sind gespannt, Ihre Meinung zu unseren Bemühungen zu erfahren. Wie Sie sehen, verhallt Ihre konstruktive Kritik bei uns nicht ungehört. Beteiligen Sie sich auch weiterhin an der Gestaltung Ihres 64'er-Magazins.

(aa)

Zunächst soll vorausgeschickt werden, daß sich die folgenden Produktvorstellungen, mit Zeit-, Händler- und Preisangaben hauptsächlich auf den amerikanischen Markt beziehen. Dort werden die meisten Programme und angekündigten Hardwareerweiterungen gegen Ende des Sommers oder gar erst im Herbst lieferbar sein. Welchen Wert hat dies nun für den europäischen und speziell für den deutschen Anwender? Nun er ist zunächst darüber informiert, welche neuen Produkte demnächst auf den deutschen Markt kommen werden, und kann daraus Hilfen für seine Kaufentscheidung beziehen. Leute mit Beziehungen besorgen sich diese Produkte aber auch gleich direkt aus den USA. Genug der Vorrede, was gab es Neues?

Da wären zunächst Floppy-Laufwerke als Alternative zum VC 1541, ein Thema das sicherlich vielen förmlich unter den Fingernägeln brennt. Von MSD Systems gibt es die beiden Einzel- und Doppel-Laufwerke MSD Super Disk Drive I und II (Bild 1). Formatieren, Kopieren und Verifizieren benötigen weniger als zwei Minuten, das ist also 20 mal schneller als auf der VC 1541. Die beiden Laufwerke haben vier beziehungsweise sechs KByte Pufferspeicher (das bedeutet, mehr Files können zur selben Zeit geöffnet sein), zeigen keine thermische Überlastung und lassen sich sowohl über den seriellen als auch über den parallelen Bus (User-Port, IEEE-kompatibel; also auch für die größeren System geeignet) betreiben. Der Preis: 399 beziehungsweise 695 Dollar.

Ein Einzel-Laufwerk, daß bis zu zehnmal schneller ist, bietet Concorde mit dem C-321P (Bild 2) für 389 Dollar. Dieses Laufwerk ist allerdings nur über den User-Port zu betreiben. Beachtenswert ist ein Jahr Vollgarantie.

Speziell für den seriellen Bus wurde das Einzel-Laufwerk Commander II von Telesys entwickelt. Die Zeiten liegen in derselben Größenordnung wie bei der VC 1541. Dieses Laufwerk soll sich aber besonders durch die hohe Zuverlässigkeit vom Orginal-Laufwerk unterscheiden, und hat ebenfalls ein Jahr Garantie. 369 Dollar kostet der Commander II.

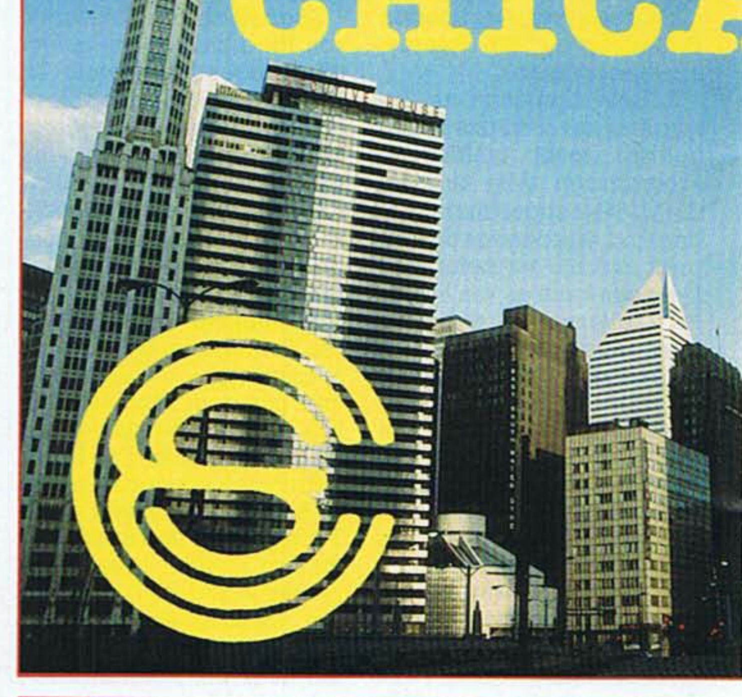






*Bild 1. Die Alternativ-Laufwerke von MSD, Super Disk Drive I und II.*



*Bild 2. Das Laufwerk C-321P von Concorde für 389 Dollar ist bis zu zehnmal schneller als die Floppy VC 1541.*

## Superfarben

Einen neuen Commodore-kompatiblen Farbdrucker hat Okidata mit dem Okimate 10 (Bild 3) vorgestellt. Dieser Thermo-Matrixdrucker bringt im Einfarben-Modus bis zu 136 Zeichen in eine Zeile mit einer Druckgeschwindigkeit von bis zu 60 Zeichen in der Sekunde. Einen Beispielausdruck der brillianten Wiedergabe mit allen auf dem C 64 möglichen Farben zeigt Bild 4.

Auch Epson bringt mit dem 160 CPS ( characters per second) schnellen JX-80 Drucker Farbe in den Ausdruck. Der JX-80 wird mit der obligaten Centronics-Schnittstelle rund 1500 Dollar kosten.

Für die Ungeduldigen bietet Telesys einen 99 Dollar teueren Drucker-Puffer Turboprint/GT (zu erkennen im Bild 1) mit wahlweise 16 (99 Dollar) oder 32 (129 Dollar) KByte Speicher für die Verbindung von Commodore 64 mit Centronics-Druckern an. Damit lassen sich auch alle Grafikzeichen des C 64 übertragen.

Speziell für die Erstellung von hochauflösenden Grafiken wurde das Super Sketch (Bild 5) von Personal Peripherals entwickelt. Es handelt




**Die »Consumer Electronic Show« findet jedes Halbjahr in den USA statt. Auf der »Summer CES« in Chicago wurde ein Phänomen besonders deutlich, die Dominaz des Commodore 64 auf dem Heimcomputer-Markt. Software und kein Ende war die Devise. Dabei wurde fast jede neue Programm-Entwicklung zunächst auf dem C 64 vorgestellt und auf anderen Computern zu einem späteren Zeitpunkt in Aussicht gestellt. Aber auch bei der Hardware gab es für den C 64 einiges an Neuigkeiten.**

*Bild 3. Der neue Farb-Thermodrucker Okimate 10.*

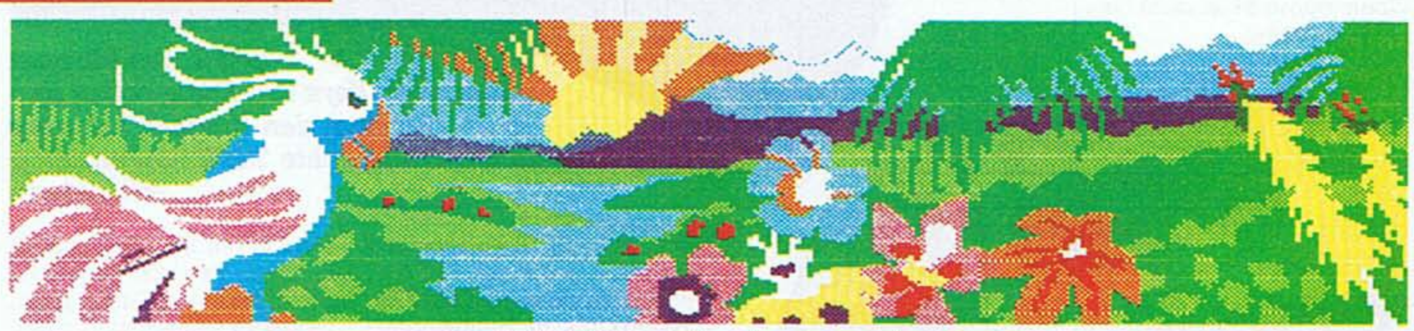


*Bild 4. Dieser Ausdruck wurde mit dem Okimate 10 von Okidata in Verbindung mit dem C 64 erstellt.*

sich dabei um eine Kombination von Grafiktablett und »CAD-Eingabetisch«. Die Übermittlung der Zeichenkoordinaten erfolgt nicht durch eine sensitive Fläche, sondern mit Hilfe eines beweglichen Plastikarms. Die Auflösung beträgt 160 x 200 Punkte und der Preis 60 Dollar. Beachtenswert ist die reiche Auswahl an Software, wie beispielsweise der Master Home Planer für die schnelle Planung von Gebäudegrundrissen und Zimmereinrichtungen oder die Musik Box zum Komponieren auf dem Grafiktablett.

In dieselbe Bresche wie das Koala Pad versucht auch der Digitiser (Bild 6) von dem Drucker- und Expansionhersteller Cardco zu schlagen. Mit etwas besser Auflösung aber mit weniger Software. Koala Technologies selbst ist mit seinen Muppet Learning Keys (Bild 7) in Richtung »Vereinfachung der Eingabe für Kinderhände« gegangen. Dieses Eingabemedium für Kinder im Alter zwischen drei und sechs Jahren soll denen das Alphabet, die Zahlen, die Farben und Formen für 80 Dollar näherbringen.

## Mit Licht arbeiten

Einen Schwerpunkt bei den neuen Produkten stellten die Präzisions-Lichtgriffel dar. Für nicht weniger als 250 Dollar ist der Gibson Lichtgriffel zu haben (die Vermarktung erfolgt durch Koala). Dieser doch recht hohe Preis schließt jedoch zwei Disketten, die mit sehr brauchbarer Software vollgepackt sind, ein. Diese besteht aus dem PenPainter für Freihandzeichnungen, dem PenDesigner für technische Zeichnungen mit vorgegebenen Formen (Bild 8), dem PenAnimator für Animationen auf dem C 64 und dem PenMusician für die Einführung in die Musikkomposition mit dem Lichtgriffel.

Von Inkwell gibt es ebenfalls einen Präzisions-Lichtgriffel mit einem umfangreichen Softwarepaket »Flexidraw« (Bild 9). Zusätzlich zu den oben erwähnten Programmen ist auch noch mit Transgraph ein Programm zur Übertragung der mit Flexidraw erzeugten Grafiken via Modem eingebaut und mit der Penware Utility Disk I ein Programm für die Ausgabe der Bilder auf Printer-Plotter erhältlich. Der Preis für Flexidraw ist mit 150 Dollar (einschließlich Lichtgriffel) als günstig zu bezeichnen.

## Spaß am Griff

Von Wico gab es eine ganze Reihe neuer Joysticks. So zum Beispiel den Dreiweg-Joystick mit auswechselbaren Griffen für 33 Dollar oder den »Grip Handle« mit extrem kleiner Grundfläche zum besseren Halten mit einer Hand für 25 Dollar. Es lassen sich nur die horizonta-

len und vertikalen Richtungen oder wahlweise auch die Diagonalen (zu)schalten.

Nahezu unverwüstlich soll der Prostick II und III von Newport Controls sein. Der Griff ist aus massivem Edelstahl und das Gehäuse aus unzerbrechlichen Kunststoff. Es gibt Feuerknöpfe sowohl für Rechts- als auch für Linkshänder. Stolze 25 Dollar kostet das Stück.

Als Zubehör gibt es sogenannte »Boards«, oft richtige Holzbretter in die die Joysticks der Atari- und Wico-Baureihe verrutschsicher eingesteckt werden können. Preis bei Thompson 10 Dollar.

◄ *Bild 5. Das Super Sketch Board von Personal Peripherals.*

## Sprache rein und raus

Sprachausgabe auf dem Commodore 64 ist nicht neu,

▲ *Bild 7. Die Muppet Learnings Keys von Koala sollen den Kindern den Umgang mit dem C 64 durch lerngerechte Tastaturanordnung erleichtern.*



*Bild 6. Der Digitiser von Cardco.*

es gibt bereits in Deutschland einige auf dem Markt (Speech Design, SAM). Lernsoftware mit integrierter Sprachausgabe hingegen ist neu. Comm Data stellte mehrere Rechenprogramme für Kinder vor, bei denen die Aufgaben durch einen speziellen Software-Synthesizer auch hörbar gemacht werden.

Als kleines Wunder bezeichnete ENG Manufacturing den von ihnen vorgestellten Prototyp für die Spracheingabe »Chirpee« (Bild 11). Bis zu 400 Wörter lassen sich damit erfassen. Dabei ist es gleichgültig, in welcher Sprache gesprochen wird, da diese Wörter vorher in einer Erkennungsphase dem System (C 64, Software und einem kleinen Gehäuse mit Mikrofon) praktisch erst beigebracht werden. Erstaunlich dabei ist, daß »Chirpee« auch Wörter — in einem gewissen Rahmen — versteht, die von anderen gesprochen werden, obwohl diese Leute nicht an der Erkennungsphase beteiligt waren. Die Zuverlässigkeit bei einzelnen läßt sich durch Mehrfachnennungen auf bis zu 99 Prozent steigern. 179 Dollar soll die Spracherkennung kosten.

Von professionellen Keyboards bis zu Heimwerker-Lösungen reichte die Palette bei den Musikanbietern. Melodian zeigte ein Keyboard (200 Dollar) im moderen Design mit 37 Tasten und 3 Oktaven das von von einer elektronischen Orgel kaum zu unterscheiden ist. Die Musik macht allerdings der Computer. Drei verschiedene Softwarepakete (jeweils 40 Dollar) sind für den C 64

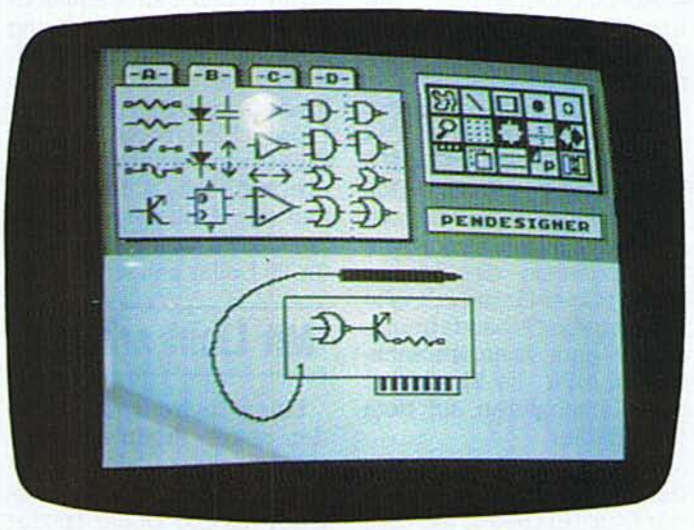



*Bild 8. Der Gibson Lichtgriffel mit dem PenDesigner*

von Melodian vorgestellt worden: ConcertMaster (Bild 12) ein Programm für die gehobenen Ansprüche, MelodyMaster zum Erlernen der Musik und der RhythmMaster der vor einem vor allen den perfekten Rhythmus beibringen soll.




*Bild 9. Mit dem FlexiDraw von Inkwell läßt sich problemlos auf der Pixelebene arbeiten.*

Bei allen drei Programmen sind die eben gespielten Noten sofort auf dem Bildschirm zu sehen. Es können aber auch gespeicherte Stücke nachgespielt werden, wobei die Programme jeden gemachten Fehler anzeigen.

Sight & Sound stecken eine eigene, kleine Tastatur für 40 Dollar (Bild 13) auf den Commodore 64, um den Kostenaufwand zu reduzieren und um dem Musiker sofort die Möglichkeit zu geben, mit dem Spielen anzufangen. Der Jazzmusiker Ryo Kawasaki stellte seine für Sight & Sound entwickelte Musiksoftware Kawasaki Synthesizer (50 Dollar) und Rythm Rocker (etwa 40 Dollar) selbst vor. Diese Musikprogramme sind mittlerweile so ausgereift, daß fast kein Unterschied mehr zu einem digitalen Sythesizer bemerkbar ist.

## Die Softwareflut

Das Angebot an neuen Software-Produkten war fast unüberschaubar. Beachtenswert dabei, fast alle Softwarehersteller zeigten ihre neuen Programme zunächst auf dem Commodore 64 mit der Ankündigung, diese Programme würden zu einem späteren Zeitpunkt auch für andere Computer wie dem PCjr, dem Aplle IIc oder den Atari-Systemen erhältlich sein. An jedem Stand war also mindestens ein C 64 vertreten, auf dem ein neues Programm lief. Ein Trend war besonders deutlich, die Hinwendung zur Lernsoftware und Spielen mit »eingebautem« Lerneffekt. Die Action- und Arcadespiele sind zwar nicht tot, wurden aber in ihrer Bedeutung etwas zurückgedrängt.




*Bild 11. Spracheingabe ist nun auch mit dem C 64 möglich. »Chirpee« versteht bis zu 400 Wörter in jeder Sprache.*




*Bild 10. Sowohl für Rechts- und Linkshänder geeignet.*

Einige Beispiele für die neue Generation der Lernprogramme sollen zeigen, was uns demnächst erwartet. Von Imagic stammt das Programm Injured Engine (Bild 14). Mit Injured Engine soll man die Funktionsweise eines Ottomotors kennenlernen. Dazu kann man sich die einzelnen Funktionsblöcke wie Kraftstoffzufuhr oder Elektrik einzeln oder im Verbund betrachten und erklärenden Text abrufen. Im nächsten Schritt wird irgendwo ein Fehler eingebaut, den es durch Diagnose und Auswechseln bestimmter Teile zu lokalisieren gilt. Man kann aber auch selbst Fehlfunktionen in einzelnen Blöcken auslösen und sich die Auswirkungen auf die anderen Elemente ansehen.

Ein ähnliches Produkt ist Cell Defense (Bild 15) von HES. Hier muß man sich als menschliche Zelle mit seinem Immunsystem den Attacken von Bakterien und Viren erwehren. Ganz nebenbei lernt man den Zellaufbau, die verschiedenen Arten von Krankheitserregern und deren Auswirkungen auf den Menschen kennen. Aber auch Faktoren wie Alter, Streß, Drogen oder Alkohol und deren Einfluß auf eine Zelle, einen Zellverbund oder den Menschen werden berücksichtigt.

Mit Type»n«Write (Bild 16) wurde nicht nur ein Textverarbeitungssystem für Kinder von HES vorgestellt. Damit ist auch ein Tastaturlernprogramm verknüpft, das von ersten Schritten mit der Commodore-Tastatur -in Form von attakierenden Buchstaben und Zahlen — bis zu perfekten Schreibmaschinenkenntnissen hinführt.

Auch bei CBS Software hat man sich nach der Pleite mit den Arcadespielen voll auf die Lernsoftware gestürzt. Das Motto von CBS lautete: »Von der Sesam Straße bis zur Wall Street«. Dies weist auch schon auf die Zielrichtung der mittlerweile 47 Programme hin; Kinder im Vorschulalter mit einfachen und Studenten sowie Autodidakten mit gehobenen Lernprogrammen zu versorgen. »Forcast« (Bild 17) soll zum Beispiel angehenden Meteorologen ein Hilfsmittel für das bessere Verständis der Wettervorhersage sein.

Aber auch die Actionspiele selbst haben eine Veränderung erfahren. Als Beispiel soll das Spiel Karate (Bild 18) von Broderbund dienen. Das Spiel zeichnet sich vor allen durch seinen Zeichtrick-Effekt aus. Der Karateka wird dabei vom Spieler mit dem Joystick an die einzelnen Gegner herangeführt und muß sich gegen



Bild 12. Das professionelle Keyboard und der ConcertMaster von Melodian.




Bild 13. Die aufsteckbare Tastatur und der Musik Processor von Sight & Sound.

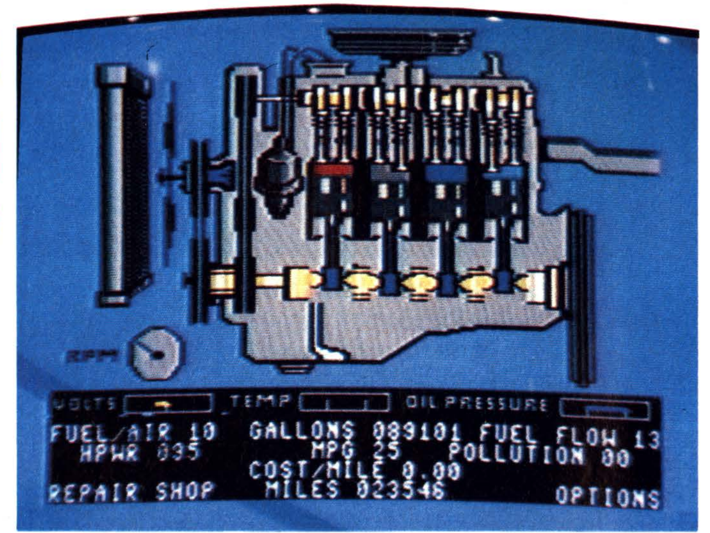



*Bild 14. Injured Engine ist ein ausgezeichnetes Lernprogramm von Imagic, um die Funktionsweise eines Ottomotors kennenzulernen.*

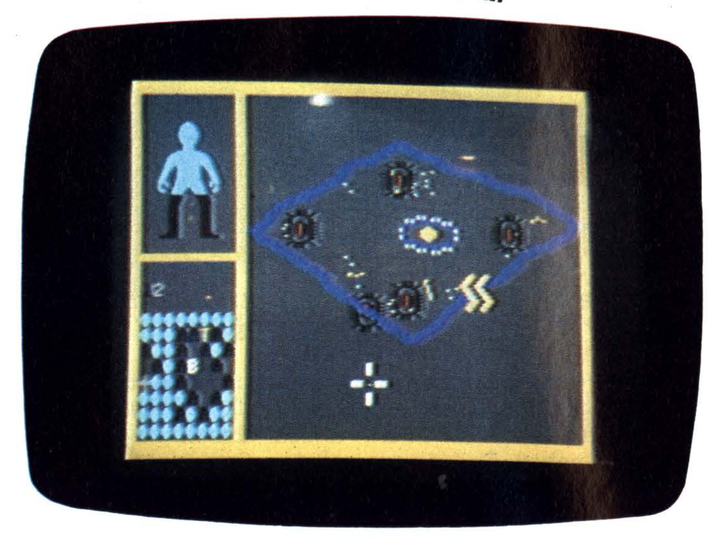


*Bild 15. Cell Defense von HES lehrt die medizinischen Grundkenntnisse des Immunsystems.*

die Braun- und Scharzgurte mit gekonnten Schlägen und Ausweichmanövern wehren.

Die Abenteuerspiele werden immer beliebter und entsprechend ist die Reaktion der Hersteller darauf. Mittlerweile braucht man nicht mehr gegen Gnome und Drachen zu kämpfen, sondern es werden richtige Szenen aus dem Leben genommen. Bei »Dallas Quest« (Bild 19) von Datasoft kann man seine Fähigkeiten gegen den Fiesling J.R. einsetzen. Aber auch Comic-Figuren dienen als Vorlage. So hat Commodore Software den »Hulk« (Bild 20) in ein Abenteuerspiel umgesetzt.

Ein beliebtes Thema ist auch der Zehnkampf geworden. Summer Games von Epyx, Declathon von Activision oder 6 Olympic Games (Bild 21) von HES sind momentan die drei konkurrierenden Programme. Sie unterscheiden sich mehr oder weniger lediglich in der Auswahl der Wettkampfarten. Gemeinsam ist bei allen drei die hohe Spielmotivation.

Bei den Anwenderprogrammen dominierten vor allen Dingen Textverarbeitung und Datenbanken. Dazu zählten der PaperClip und der Consultant von Batteries Included, der Bank Street Writer von Broderbund Software, der Creative Writer von Creative Software, das Supertext von Muse Software, Homeword von Sierra on line, der Data Manager von Timeworks, der OmniWriter von HES und und ... Zu vielen dieser Textverarbeitungprogramme gibt es sogenannte Spell Checker. Diese ermöglichen eine Überwachung der Rechtschreibung, allerdings nur im Englischen, dafür aber von bis zu 30000 oder 40000 Wörtern. Doch die Zielrichtung ist vorgegeben, die Realisierung auf dem deutschen Markt läßt aber noch auf sich warten. Interessant für uns dürfte auch Kwik-Load (Bild 22) von Datamost sein. Damit lassen sich Programme drei bis fünf mal schneller laden.

Bei Commodore selbst gab es wieder einmal eine Umbennung. Der C 264 heißt in Zukunft Plus/4. Warum dies? Nun für den C 264 wurden das Textverarbeitungs- das Grafik- das File- und das Tabellenkalkulationsprogramm wahlweise als Option angeboten. Das heißt man konnte sich für das eine oder andere Programm entscheiden. In den Plus/4 sind diese vier fest eingebaut und können die Daten interaktiv austauschen. Ansonsten ist der Plus/4 völlig identisch mit dem C 264. Eine Bemerkung am Rande: Diese vier Programme die von Tri Micro für Commodore entwickelt wurden, sind bei Tri Micro auf Diskette auch für den C 64 erhältlich. Dieser Umstand, daß neue Programme für den Plus/4 auch gleichzeitig



Bild 16. Das Kennenlernen der Tastatur in ein Weltraumspiel eingebunden. Type'n'Write von HES.




Bild 17. Die Wettervorhersage in den USA selbst nachvollziehen. Angehende Meteorologen können auch bestimmte Wettersituationen durch entsprechende Hochs und Tiefs selbst bestimmen.



*Bild 18. Karate von Broderbund. Der Zeichentrickeffekt ist faszinierend.*




*Bild 19. Versuchen Sie Ihr Glück gegen das Ekel J.R. in Dallas Quest von Datasoft.*

für den C 64 angeboten werden, zeigt einerseits deutlich, daß der Plus/4 den C 64 nicht ablösen soll, sondern lediglich nach oben hin ergänzen. Anderseits bedeutet dies für den potentiellen Käufer keine Komplizierung seiner Entscheidung. Dem C 64 steht ein langes Leben bevor. Vom C 364 wurde übrigens Abstand genommen. Er wird wohl vorläufig nicht auf den Markt kommen.

Meine amerikanischen Kollegen waren relativ überrascht, als sie hörten, daß Commodore die neue Hardware bereits auf der Hannover-Messe vorgestellt hat. Da wären zunächst einmal der Typenraddrucker DPS 1101, der kompatibel mit dem Plus/4 sein soll. Der MPS 802 Matrix-Drucker mit einer Druckgeschwindigkeit von 60 Zeichen pro Sekunde. Er hat hat in etwa dieselben Leistungsmerkmale wie der VC 1526. Der MCS 801 Farbmatrix-Drucker mit 30 Zeichen pro Sekunde und acht vertikalen Punkten sowie 640 Zeilen, sowie der Billigdrucker MPS 803, welcher wohl als Ersatz für den VC 1515 auftreten soll. Für den C 16 (der wird auch weiterhin so heißen) gibt es eine eigene Datasette, die 1531. Und letztendlich der neue Farbmonitor CM 141, der so neu eigentlich auch wieder nicht ist. Es basiert auf dem VC 1701/1702 und wurde lediglich einer kosmetischen Korrektur zur Designanpassung an den Plus/4 unterzogen. Die Angaben für die Verfügbarkeit selbst für den amerikanischen Markt sind von Commodore wieder einmal sehr schwammig: in der zweiten Hälfte des Jahres 1984. Dies ist aber ein altbekanntes Leid aller Firmen; man muß neue Produkte regelmäßig auf den großen Messen ankündigen, um im Gespräch zu bleiben, die Lieferzeiten stehen dann auf einem anderen Blatt. Die Preise und die Einführung auf dem europäischen Markt sind also mehr oder weniger noch Zukunftsmusik.

Commodore engagiert sich jetzt verstärkt im Bereich Software. Es findet dabei ein Abkehr von der bisher verfolgten Linie, billige und deswegen auch relativ einfache Software anzubieten. Commodore scheint von nun an auf Qualität zu setzen. Durch ein Abkommen mit der Marvel Comics Group werden deren Comic-Helden in Abenteuerspiele umgesetzt. Typische Beispiele sind die Questprobe-Serien Hulk und Spiderman. Dabei sollen die Comic-Bücher von Marvel simultan mit den Abenteuerspielen verkauft werden. Weitere Produkte in der Abenteuer-Serie sind Satan's Hollow und Solar Fox.

Auch bei den Lernprogrammen gibt es von Com-

modore viel Neues zu berichten. Mit Just Imagine können Kinder ab vier Jahren ihre eigenen Animationsgeschichten kreieren. Mit Number Builder müssen die Kinder eine bestimmte Zahl durch Addieren, Subtrahieren, Multiplikation oder Division erhalten. Für alle die Spaß an der Astronomie haben, gibt es Window to Galaxies. Ein Programm, das von jedem Punkt der Erde zu jedem Zeitpunkt die Sternekonstellation zeigt. Die Positionen von mehr als 12000 Sternen, von 88 Sternbildern, der Sonne, des Mondes, von acht Planeten, von Kommeten und Galaxies können auf dem Bildschirm dargestellt werden. In der Productivity-Rubrik stach besonders das Grafikprogramm B/Graph (Bild 23) hervor.

Auf der CES ist es üblich, die besten Programme zu küren. Diese finden sich dann an den verschiedenen Ständen mit den in Amerika so beliebten Auszeichnungen oder Orden wieder. Diese Software Showcase genannte Prämierung betraf nur Programme für die wichtigsten fünf Heimcomputer: Apple II/IIe, IBM-PCjr, Atari Home Computer, ColecoVision/Adam und mit weiten Abstand vorne, Commodore 64.

Gekürt wurden Beach-Head von Access Software, Matchboxes von Broderbund Software, Jack Attack, Micro Cookbook und International Soccer von Commodore, Childpace von Computerrose, Dragonhawk, In the Chips und Moondust von Creative Software, Music Construction Set von Electric Arts (ECA), Fax und Summer Games von Epyx, OmniWriter/Omnispell, Cell Defense, HES Games und Turtle Toyland,Jr. von Human Engineered Software (HES), The Rug Rider von International Tri Micro, Super-Text und Space Taxi von Muse Software, 64 Doctor von Practicorp, Superbase 64 von Precision Software, Falcon Patrol von Quicksilvia, Pogo Joe von Screenplay, Musicomp-Music Processor von Sight & Sound International, Model Diet, Computer Mechanic und Dancing Feats von Softsync, Flight Simulator II von Sublogic, Campain'84 von Sunrise Software, Data Manger 2 und Dungeon of the Algebra von Timeworks sowie schließlich MusiCalc von Waveform Corporation. Diese Programme wurden also von einer unabhängigen Jury zu den Rennern auf dem Commodore 64 erklärt.

Der Commodore 64 hat eine überragende Marktstellung gewonnen. Die neueste und beste Software wird auf dem C 64 vorgestellt, die anderen Systeme müssen warten, bis die Programme umgeschrieben sind. Bisher war das noch umgekehrt (vor allem was die Atari-Computer betrifft). Das Hardware-Angebot auch von unabhängigen Herstellern wird immer umfangreicher. Andere, neue Computer werden einige Schwierigkeiten haben, sich gegen die Dominanz des C 64 durchzusetzen. Von MSX-Systemen war zwar schon einiges zu sehen, aber der große Durchbruch ist noch lange nicht in Sicht. Es bleibt abzuwarten, wann es den ersten C 64-kompatiblen Computer (ähnlich wie beim IBM PC) auf dem Markt gibt. (aa)



**Bild 20. Hulk von Commodore lehnt sich an die Comic-Serien von Marvel an.**

**Bild 21. Hes Games**

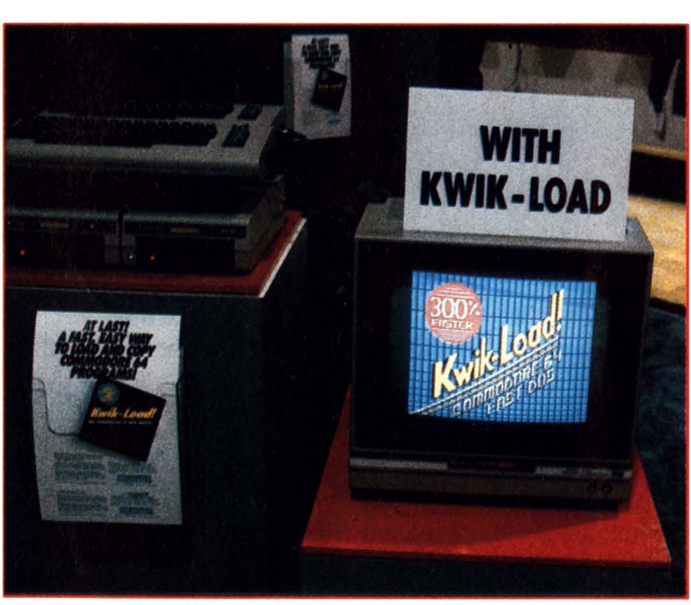


*Bild 22. Kwik-Load von Datamost*

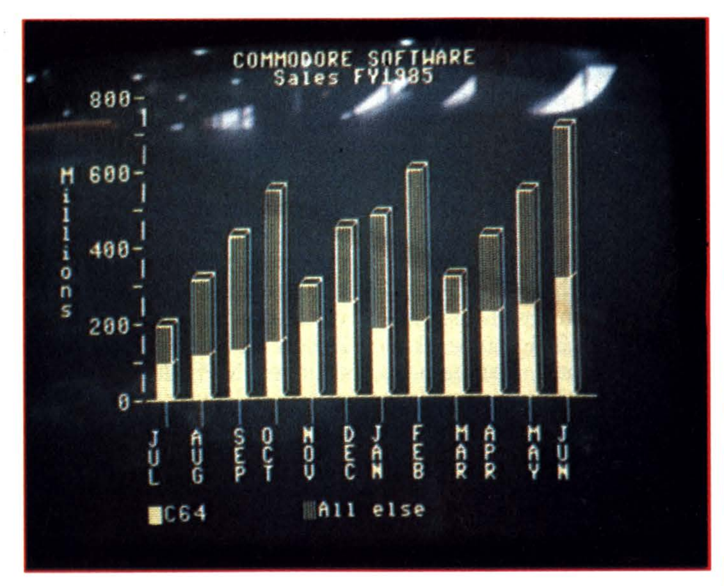


***Bild 23. B/Graph von Commodore vereinfacht die Erstellung von dreidimensionalen Blockgrafiken ungemein.***




## Steuern mit dem User-port?

*Ich möchte über den Userport einen x/y/z Bohrfrästisch steuern. Wie ist so etwas zu realisieren?*

*Thomas Wistuba*



## Probleme mit Monitoren

*Wie schließe ich einen Monitor an den VC 20 an? Anfragen bei Händlern ergaben nur, daß ein entsprechendes Kabel anzufertigen sei, aber nicht wie. Wer kann helfen?*

*Inge Märkle*

*Ich besitze einen Commodore SX 64 und ein Sony Farb-TV ohne Video-Eingang. Wie kann ich diese Geräte zusammenschalten, damit der Farbfernseher als Monitor genutzt werden kann?*

*Hans-Jürgen Mundstock*

*Wie bekomme ich die streifenförmig auftretenden Farbränder auf dem Fernsehschirm weg?*

*Gibt es eine Möglichkeit, aus dem C 64 ein RGB-Signal zu holen?*

*Peter Stahn*

*Ich besitze einen Blaupunkt-Farbfernseher mit RGB-Eingang. Welche Firma bietet ein RGB-Interface (bzw. entsprechenden Schaltplan) für den C 64 an und zu welchem Preis?*

*Gernot Murawski*

*Wie kann ich mit dem Video-Recorder Panasonic N8003T das Signal vom Computer aufnehmen? Der Fernseher vom Typ Saba T5658 und 75 Ohm-Kabel funktioniert einwandfrei.*

*Eugen Anger*

*Wie schließe ich den C 64 an einen Fernseher mit Scart-Buchse an (Nordmende 1534, 3540; Loewe)?* *Walter Wagner*

## Anschlußprobleme mit Druckern

*Wie kann man die KB 50 Tastatur des Brother HR15 an den Userport des C 64 anschließen, um für Textverarbeitungssysteme eine DIN2137-Tastatur zur Verfügung zu haben?*

*Horst Singer*

*Ich habe eine Schreibmaschine Brother CE-50 mit seriellem Interface (Adresse 4), die mit meinem C 64 gut zusammenarbeitet. Nur mit Datamat von Data Becker kann ich keinen Ausdruck erstellen. Die Firma hat mir auf schriftliche Anfrage nur nichtssagend geantwortet. Wer kann helfen?*

*Gerd Mandelkow*

*Kann man die Kugelkopf-Brother 7900 (durch einen Trick?) über irgendein Interface an den C 64 anschließen?*

*Stefan Wülfert*

*Gibt es eine softwaremäßige Verbindung zwischen dem C 64 und Brother CE-60?*

*Manfred Mattern*

*Wie kann ich die Sonderzeichen der Schreibmaschine Privileg 3000 ansprechen und ausdrucken? Welchen Code hat die KB-II-Taste?*

*Dr.Ernst Suchalla*

*Ich habe einen Fernschreiber Siemens T100 und suche einen Anschlußplan für den C 64.*

*Burghard Schmolling*

*Wer kennt einen Hard- oder Software-gesteuerten Anschluß einer Olivetti Praxis 30 an den C 64?*

*Josef Hauk*

## Programmunterbrechung bei Druckerausgabe

*Meine Hardware: VC 64, Floppy 1541, NEC Drucker 8023 B-N mit Interface 9200 für VC 64. Bei Ausgabe über Drucker von mir erstellter Programme, als auch gekaufter Programme (zum Beispiel Datamat von Data Becker) erfolgt nach unterschiedlich langer Druckerausgabe die Fehlermeldung »Device not present error«. Manchmal funktioniert die Druckerausgabe (mit Endlosblatt) seitenweise und unterbricht dann plötzlich mit oben angegebener Fehlermeldung; das andere Mal werden nur ein paar Zeilen ausgegeben und wieder erfolgt die obige Unterbrechung. Natürlich sind alle Geräte (C 64, Floppy, Drucker) eingeschaltet und betriebsbereit. Handelt es sich hier nur um einen Hardware- oder Softwarefehler? Kann mir hierzu jemand helfen?*

*(Rudolf Ott)*

## Hardcopy mit NEC 8023B-N

*Wer bietet Software an, um mit C 64 und Drucker NEC 8023B-N hochauflösende Grafik und Hardcopys zu erstellen?*

*Wolfgang Jaworski*

### Fragen Sie doch!

Selbst bei sorgfältiger Lektüre von Handbüchern und Programmbeschreibungen bleiben beim Anwender immer wieder Fragen offen. Viel mehr Fragen ergeben sich bei Computer-Interessenten, die noch keine festen Kontakte zu Händlern, Herstellern oder Computerclubs haben. Sie können der Redaktion Ihre Fragen schreiben oder Probleme schildern (am einfachsten auf der beigehefteten Karte). Wir veranlassen, daß die Fragen von einem Fachmann beantwortet werden. Allgemein interessierende Fragen und Antworten werden veröffentlicht.

## Bits hörbar machen?

*Ist es beim C 64 möglich, beim Laden von Datasette die Signale hörbar zu machen?*

*Werner Frings*

## Simons Basic mit Turbo-Tape?

*Wer kennt eine Möglichkeit, Simons Basic und Turbo-Tape gleichzeitig benutzen zu können?* *Rolf Lehr*

## Knisternde Bits?

*Mein C 64 gibt nach dem Einschalten knisternde Geräusche über den Fernseher von sich, die allerdings mit der Zeit wieder verschwinden. Es liegt kein Wackelkontakt am Kabel vor. Wer weiß Rat?*

*Matthias Brunke*

## Text und Grafik mischen?

*Ich besitze einen Commodore 64 und bin gerade dabei, ein Adventure zu schreiben. Nun stehe ich vor einem Problem. Wie kann ich in Simons Basic hochauflösende Grafik und Text gleichzeitig darstellen? Ich benötige nur ein kleines Textfenster für die Eingabe von Antworten.* *Frank Schager*

## Wer kennt »Aztec Tomb«?

*Ich besitze das Spiel »Aztec Tomb« für den C 64 und komme einfach nicht weiter. Nachdem ich am Bullen vorbei durch das Tor gehe, stehe ich vor einer Pflanze auf »Waste Ground«. Ich trage bei mir: old chest, red clock, dead mouse, door-key. Wer kann mir weiterhelfen?*

*Stephan Sache*




## Hier gibt es Clubs

Der ICC hat derzeit 26 Mitglieder, ausschließlich C 64-User mit Datasette, 1541 Floppy, 1515/1526/Epson FX-80 Drucker. Er beschäftigt sich schwerpunktmäßig mit allen Aspekten des List- und Kopierschutzes, Modemtechnik, Hilfestellung für Einsteiger, Softwaretausch, Hardware-Sammelbestellun gen, Förderung von Kommunikation unter den Mitgliedern auf allen möglichen Ebenen. Treffen gibt es derzeit noch nicht, sind aber mittelfristig geplant.

Geplant ist für die nahe Zukunft eine eigene Mailbox. Die Entwicklung von dazu nötiger Betriebssoftware ist ein weiterer Arbeitsschwerpunkt. Der Club freut sich in diesem Zusammenhang über jegliche Form der Mithilfe. Neue Mitglieder sind natürlich jederzeit willkommen.

Der Club gibt monatlich ein 12seitiges Info für seine Mitglieder heraus. Der Jahresmitgliedsbeitrag ist 60 Mark, eine dreimonatige Probemitgliedschaft für 20 Mark ist möglich. Ein Probeinfo gibt es für 5 Mark.

INTERFACE-COMPUTER-CLUB, Hindenburgstr. 98, D-2120 Lüneburg, den 22/06/84.

## Software per Telefon

Wir sind ein C 64-Userclub, der sich seit einiger Zeit intensiv mit der neuen Technik der DFÜ per Telefon beschäftigt.

In unserem monatlichen Clubinfo berichten wir ausführlich über DFÜ und die damit zusammenhängenden Probleme — natürlich vornehmlich auf den C 64 bezogen.

Wir planen, ab Mitte Juli unseren Mitgliedern per Telefon Software zu schicken. Ein Terminal-Programm, das den C 64 in ein intelligentes Terminal verwandelt, wurde bereits von uns erstellt.

Speziell für DFÜ-Neulinge haben wir ein DFÜ-Sonderinfo mit allen Informationen, die man braucht um mit dem C 64 online zu kommen, herausgegeben. Das Info enthält auch eine Beschreibung unseres Terminalprogrammes. Interessierte können es gegen 3 Mark in Briefmarken bei uns bestellen. Anfragen bitte an unsere Clubadresse oder telefonisch.

Andreas Voigt
INTERFACE-COMPUTER-CLUB, 2120 Lüneburg, Hindenburgstr. 98, Tel.: 04131/37876.

## Adreß- und Telefonregister aus Ausgabe 5 mit Datasette?

*Ich möchte gerne »Adress- und Telefonregister« aus der Ausgabe 5 mit Datasette verwenden, weiß aber als totaler Anfänger nicht, wie ich das Programm umschreiben muß. Ist dies überhaupt möglich?*

*Dieter Plaum*

Mit einigen Änderungen läßt sich das Programm auch mit Datasette verwenden:

```
990 ...:OPEN 1,1,0
1050 ...:OPEN 1,1,1
2040 ...:OPEN 1,1,1
```

Bei dieser Version funktioniert auch das »SUCHEN« und die »AUSGABE SORTIERT«.

Jürgen Zier

## Basic-Zeilen größer 80 Zeichen?

***Die maximale Zeilenlänge beim C 64 beträgt 80 Zeichen. Bei manchen Programmen werden diese teilweise überschritten. Wie kann man das Programm trotzdem zum Laufen bringen?***

***Rainer Jost***

Diese vor allem von Anfängern häufig gestellte Frage läßt sich einfach beantworten. Im Handbuch des C 64 befindet sich im Anhang D auf Seite 130/131 eine Tabelle der Abkürzung von Basic-Schlüsselwörtern. Fast jeder Basic-Befehl läßt sich durch Eingabe des ersten Buchstabens, gefolgt vom geshifteten zweiten Buchstaben des Befehls abkürzen. Zum Beispiel wird aus der Eingabe von P (Shift)O das Wort POKE, oder P (Shift)R wird zu PRINT # im Gegensatz zu »?« für PRINT. So kann eine Basiczeile beim Listen 80 Zeichen überschreiten. Versuchen Sie nicht, diese Zeile hinterher zu verändern, indem Sie mit dem Cursor in diese Zeile gehen und nach einer Änderung wieder mit RETURN verlassen. Dann wird automatisch der letzte Teil, der sich in der dritten Reihe befindet, abgeschnitten. Zum Editieren geben Sie am besten die komplette Zeile mit den entsprechenden Abkürzungen noch einmal ein.

## Grafik mit 1526?

*Ich besitze den 1526-Drucker von Commodore. Wie kann man ihn grafikfähig machen?*

*Percy Dahm*

Der Commodore 1526 ist hardwaremäßig nicht für hochauflösende Grafik vorgesehen, daß heißt, es besteht keine Möglichkeit einer Einzelnadelansteuerung. Allerdings kann man ein Zeichen selbst definieren und so — allerdings sehr umständlich — Grafik drucken. Dazu definiert man sich das Zeichen als geeignetes Grafiksymbol, fährt durch Ausgabe einer Reihe von Blanks an die richtige Position, druckt das Zeichen und veranlaßt dann einen Wagenrücklauf ohne Zeilenvorschub. Nun wird das nächste Grafikzeichen definiert und das ganze Spiel wiederholt sich. Mit dieser Methode ist sogar eine Grafik-Hardcopy möglich (siehe 64'er, 5/84, Seite 74: »Ein eigentlich unmögliches Programm«).

## Von Kassette auf Floppy?

***Ich möchte mir demnächst eine Floppy 1541 kaufen. Kann man mehrere Programme, die man auf Kassette hat (gekaufte Spiele und anderes), auf eine einzige Diskette speichern um Platz zu sparen? Wenn ja, wie?***

***Andreas Schmelzer***

Natürlich kann man auch mehrere Programme auf einer Diskette abspeichern. Kommerzielle Software auf Kassette ist jedoch in der Regel mit Autostart und einem Softwareschutz versehen, so daß ein Kopieren in der Regel nicht möglich ist.

### Wollen Sie antworten?

Wir veröffentlichen auf dieser Seite auch Fragen, die sich nicht ohne weiteres anhand eines gutes Archivs oder aufgrund der Sachkunde eines Herstellers beziehungsweise Programmierers beantworten lassen. Das ist vor allem der Fall, wenn es um bestimmte Erfahrungen geht oder um die Suche nach speziellen Programmen beziehungsweise Produkten. Wenn Sie eine Antwort auf eine hier veröffentlichte Frage wissen — oder eine andere, bessere Antwort als die hier gelesene — dann schreiben Sie uns doch. Die Antworten werden wir in einer der nächsten Ausgaben publizieren. Bei Bedarf stellen wir auch den Kontakt zwischen Lesern her.

## Druckerroutinen umschreiben?

*Wie kann ich Programme, die für den Drucker VC 1526 geschrieben sind und dessen Fähigkeiten ausnutzen für den VC 1525 umschreiben?*

*Bruno Schiffer*

Der VC 1526 verfügt über eine Reihe komfortabler Befehle zur formatierten Ausgabe, die auf dem VC 1525 nicht vorhanden sind. Diese Formatierungsbefehle müssen für den VC 1525 durch Software ersetzt werden. Im Prinzip geht es darum, eine Art PRINT USING-Routine zu schreiben. Am besten schreibt man alle Programmteile, die Ausgaben an den Drucker liefern, völlig neu.

## »Strubs« mit Datasette?

*Warum bringen verschiedene POKEs, die im Commodore-Basic einwandfrei funktionieren (zum Beispiel Listschutz) mit Simons Basic das Programm zum Abstürzen?*

*Sebastian Obermayr*

»Strubs« kann leider nur mit der Floppy arbeiten, da mehrere Files gleichzeitig geöffnet werden.

## POKEs im Simons Basic?

*Warum bringen verschiedene POKEs, die im Commodore-Basic einwandfrei funktionieren (zum Beispiel Listschutz) mit Simons Basic das Programm zum abstürzen?*

*Sebastian Obermayr*

Einige Adressen der Zeropage werden von Simons Basic anders belegt, so daß eine Reihe von POKE-Befehlen nicht mehr funktioniert.

## Nochmals Simons Basic

*Benötigt Simons Basic einen Teil des Basic-RAMs? Wenn ja, wieviel?*

*Lars Steubesand*

Simons Basic belegt die oberen acht KByte des Anwender-RAM, der verfügbare freie Speicher für Basicprogramme reduziert sich also um diese acht KByte.

## Resettaste für VC 20?

*Sind die Anschlüsse für eine Resettaste am Userport von VC 20 und C 64 identisch? Ist so eine Taste auch am C 64 anschließbar?*

*Adrian Schneider*

Die beiden Userports sind nicht 100% identisch, jedoch gibt es beim Anschluß eines Resettasters keine Unterschiede. Durch Verbindung von Pin 1 und Pin 3 erfolgt bei beiden Computern ein Reset.

## Raubkopien strafbar?

*Darf man gekaufte Software auf Leerkassetten kopieren und verkaufen? Oder ist das gesetzlich verboten, so daß eine Bestrafung erfolgt?*

*Maik Rogalla*

Das Kopieren und Weiterverkaufen von fremder Software ist ein Verstoß gegen das Urheberrecht und wird entsprechend geahndet. Man muß in jedem Falle mit hohen Schadenersatzforderungen der Herstellerfirmen rechnen.

## Bauanleitung für EPROM Brenner

*Wer hat eine Bauanleitung für einen EPROM Brenner für C 64? Ausgabe 4/84.*

Ich habe eine Bauanleitung zum Programmieren der ICs 2716, 2732 und 2764 entwickelt. Diese besteht aus einem Layout mit Bauanleitung plus dazugehörender Software. Die Kosten sind wegen der wenigen Bauteile sehr gering und liegen noch unter 50,— DM (o. Nullpontfassung).
Bernd Bause, Bahnhofstr. 21, 2000 Hamburg 55.

## Hilferuf

*Ich habe so wenig Programme. Bitte! Bitte! Schickt mir welche.*

*Sven Reichel*

So ergeht es wohl jedem Anfänger. Wir von der 64'er Redaktion empfehlen das 64'er Magazin als Abhilfe.

## Apple-Software für den C 64

*Wie Sie in Ausgabe 4/84, Seite 9 schreiben, ist bei Mimic Systems, Kanada ein sogenanntes AP-Modular Pak, das erlaubt Apple II-kompatible Soft- und Hardware am C 64 zu betreiben, erhältlich.*

*Leider geben Sie die genaue Adresse des Anbieters nicht an. Nun möchte ich Sie um die genaue Adresse von Mimic bitten.*

*Walter Wagner*

Hier ist die Adresse:
Mimic Systems
Pioneer Software, Inc. (# 217 — 620 View Street, Victoria, BC, Canada, V8W 176).

## Programme weg beim Reset?

*Ich habe in meinen C64 eine Resettaste eingebaut, und wenn ich diese benutze, ist das Programm weg. Wie kann ich dieses Programm wieder hereinholen? Suche dazu die PEEKs und POKEs mit Adressen.*

*Arndt Wörau*

Nach einem Reset bleiben Maschinenprogramme nach wie vor im Speicher erhalten und können wieder mit entsprechendem Sys gestartet werden. (Dies ist jedoch nicht der Fall bei Maschinenspracheprogrammen, die im Kassettenpuffer liegen [dez. 828-1019], da dieser bei einem Reset gelöscht wird.) Bei Basicprogrammen muß man schon mehr machen, um es wieder herzuholen:

POKE 2049,99: POKE 2050,99: POKE 45,X: POKE 46,Y: SYS 42291

Dabei sind X und Y die Werte, die man vor Drücken des Reset-Knopfes durch ?PEEK(45);PEEK(46) abfragen sollte. Kommt man aus Versehen auf den Knopf und weiß den Inhalt dieser Speicherstellen nicht mehr, so empfiehlt es sich, in die Speicherstellen 45 und 46 angenommene Werte für die Programmendadresse zu POKEn (45 = low Byte, 46 = high Byte).

## Ein Tip des Autors

In meinem Programm »Adreß- und Telefonregister«, das in der Ausgabe 5/84 im 64'er-Magazin veröffentlicht wurde, habe ich zur Eingabe einer Telefonnummer fünf Stellen vorgesehen. Um mehr Stellen für eine Eingabe zu ermöglichen, ist die Zuweisung der Variable 1 in Zeile 1970 zu ändern (zum Beispiel 1=8 für 8 Telefonziffern). Die nachstehenden Befehle in derselben Zeile bleiben unverändert.

Außerdem möchte ich noch auf einen »Schönheitsfehler« hinweisen: In Zeile 1060 muß es statt »for i = o to an« heißen: »for i=0 to an«. Da o allerdings im Programm nicht als Variable benutzt wird und daher immer * ist, hat dieser Fehler keine Auswirkungen auf den Programmablauf.

Bei der Eingabe der Datensätze sollte darauf geachtet werden, nicht die »f7«-Taste zu drücken, um etwa ein Feld frei zu lassen, da sonst die Datei später nicht korrekt abgespeichert werden kann. Soll ein Feld frei gelassen werden, so ist die Leertaste (und »return«) zu drücken.

Arne Weitzel

## Wer kennt Master 64?

*Ich möchte meine mit Master 64 entwickelten Programme auf 8032 übertragen und suche Informationen bzw. Anwender von Master Version 64 und 8032. Wer kann helfen?*

*Walter Hochwald*




So erging es auch mir an einem trüben Wintertag, kurz vor den ins Haus stehenden Festtagen und der damit verbundenen Freizeit zwischen den Tagen. Der erste Gang führte kurzentschlossen in einen für gewöhnlich gut sortierten Zeitschriftenladen. Nach einigem Stöbern im Berg der Computerzeitschriften verklärte sich der schon etwas trübe Gesichtsausdruck des Ladeninhabers zu seligem Lächeln, als er den Rechnungsbetrag mit spitzem Bleistift und unter heftigen Zungenbewegungen ermittelte. So war denn zumindest für ihn das Weihnachtsfest gesichert, ich dagegen erklomm mit meiner Informationslast die heimischen Treppen und fing, kaum wieder zu Atem gekommen, damit an, mein Wissen über Drucker zu vervollkommnen.

Ein rascher Blick in die schon leicht schwindsüchtige Brieftasche erleichterte den Entscheidungsprozeß erheblich, schon nach relativ kurzer Zeit stand fest, daß ein Schönschreibdrucker nicht in Frage kam. Was blieb, waren die einfachen Matrixdrucker, darunter auch der MPS 801 von Commodore, der gerade als »Neuheit« auf dem deutschen Markt erschienen war.

Mit den Informationen über drei in Frage kommende Drucker im Geist und in der Hand, begab ich mich nun auf die Suche nach einem Computershop, um das theoretisch erarbeitete Wissen praktisch zu überprüfen. Dieses Unterfangen stellt in einem computertechnischen Entwicklungsland, das diese unsere Republik nun einmal ist, ein größeres Problem dar, zumal in einer Kleinstadt wie Hamburg. Da Computerläden grundsätzlich mehrere Kilometer vom nächsten freien Parkplatz entfernt liegen — eigene Parkplätze lohnen sich wohl nicht, da die Kunden zum größten Teil Jugendliche sind — ergab sich die einmalige Gelegenheit, etliche Pfunde, die sich in mitternächtlichen Sitzungen am Computer angestaut hatten, wieder abzulaufen.

Endlich war der erste Laden erreicht, also raus mit dem bereits im Geiste geübten Satz: »Ich wollte mir mal Drucker für den C 64 angucken und ...«

Ungläubiges Staunen beim Verkäufer: »Drucker? Ham wir nich da, sind aber bestellt.«

Darauf ungläubiges Staunen beim Kunden, (»Aha«) gefolgt vom raschen Ortswechsel. Schauplatz des nächsten Dramas ist die Computerabteilung eines großen Kaufhauses. Inmitten einiger Taschenrechner fristet ein einsamer Homecomputer sein Dasein, von fachkundigem Personal vor kaufwütigen Kunden beschützt: »Das ist unser Letzter, den können wir nicht rausgeben... — aber wir haben schon neue bestellt.«

Also rein ins Gewühl. »Ham se mal'n Moment Zeit? Ich wollt mir Drucker für...«

»Karl, kommste mal, da will einer 'nen Drucker kaufen.« — »Drucker? Ham wir nich, sind aber...«

Man soll ja die Hoffnung nie aufgeben, also weitergemacht mit der Stadtrundfahrt. Und tatsächlich, im siebten oder achten Laden geschieht das Wunder: »Drucker? Hatten wir bestellt, sind gerade gekommen. Was darf's denn sein?«

Ehrfurchtsvoll bestaunt der entzückte Kunde die sauber aufgereihte Menge der Drucker: Beide nebeneinander.

Der Ladenschluß ist schon bedrohlich nahe, also wildentschlossen zugeschlagen. »Was kostet denn bei Euch der GP 500 und was der MPS 801?« — »Hä? Das ist doch ein und dasselbe Gerät, nur mit nem anderen Gehäuse...«

Aha, das stand in keiner der gelesenen Zeitungen. Mißtrauisch geworden, kommt die nächste Frage: »Der sieht aber innen drin dem GP 100 verdammt ähnlich..«

»Is ja auch das Nachfolgemodell, leiser, mit verbessertem Schriftbild und nem ganz neuen Farbbandsystem...«

Naja, lange Rede, gar kein Sinn, es

Naja, lange Rede, gar kein Sinn, es gibt viel zu drucken, packen wir's das im Autohandel. Zurück zum eigenen Fahrzeug, unter einem Arm den Karton Papier, unter dem anderen den Drucker. Nach kurzer Diskussion mit einer netten Politesse, die gerade dabei ist, die Scheibenwischer festlich zu schmücken, kann endlich die Heimfahrt durch den Dschungel des Feierabendverkehrs angetreten werden.

Zuhause angekommen, wird erstmal ausgepackt und die Betriebsanleitung studiert. Dabei kristallisiert sich immer klarer das erste Druckvorhaben heraus: Eine deutsche Fassung des Heftchens muß her. Da der durchschnittliche Mitteleuropäer neben seiner Muttersprache und diversen Computerdialekten auch noch umfassende Kenntnisse des Englischen besitzt, fällt es nicht schwer zu erraten, was mit den Anweisungen des Handbuches gemeint ist.

Schließlich sind Papier und Farbbandkassette eingelegt, der erste Versuch kann starten. Schalter auf »Test« und ab dafür. Und siehe da, der Verkäufer hat nicht gelogen, der Drucker ist vergleichsweise leise. Statt mit den Phonzahlen eines startenden Jumbos säuselt der Druckkopf mit Diskothekenniveau über das Papier.

Da wenigstens das Schriftbild den Versprechungen entspricht, kann der Test abgeschlossen werden, der Drucker wird jetzt mit dem Computer verbunden. Dies geschieht mittels des beigelegten Buskabels, das die stattliche Länge von 60 cm aufweist. Man kann sich also aussuchen, ob einem die Nadeln von links oder rechts ins Ohr knattern... Nun

**Früher oder später kommt im Leben eines Computerbesitzers der Tag, an dem er beschließt, seinen Gerätepark um den lange fälligen Drucker zu erweitern.**




ja, so wird man wenigstens zu einem kompakten Aufbau der Computeranlage gezwungen, es bleibt sogar noch Platz auf dem Tisch für Aschenbecher und eine Flasche des von ortsansäßigen Brauereien gelieferten Kühlmittels.

Nun lassen wir in Gedanken zwei Monate verstreichen, wir befinden uns im Januar des Jahres 1984. Nach 150 bis 200 Blatt Papier wird das Farbband allmählich schwächer, ein frisches muß her. Halt, da war doch was? Ach ja, im Handbuch steht was von austauschbarem Inker. Das muß das Ding sein, wo die Tinte drin ist. Da das Farbband selbst noch gut ist, tuts also auch so eine Austauschkartusche. Im Zuge der nun folgenden Stadtbesichtigung kommt insgesamt zwölfmal die Standardantwort des Fachpersonals: »Ham wir nich, is aber bestellt...«

Eine Woche später liegt das gute Druckbild in den letzten Zügen. Es hilft nichts, wenn es keine Inker gibt, muß halt ein komplettes Farbband her. Also machen wir mal etwas Neues, nämlich eine kleine Sightseeing-Tour durch die Computerläden und Kaufhäuser. Der geneigte Leser wird sich wohl schon denken können, wie die meistgegebene Antwort auf mein Kaufbegehren lautete....

Irgendwo muß es dann eine Panne in der Logistik gegeben haben, denn einige Tage später bekam ich mein Farbband, natürlich ausgerechnet in dem Laden, wo ich es nie vermutet hätte. Zwar kostete der ganze Spaß inklusive der Spritkosten weit über fünfzig Mark, aber immerhin konnte ich wieder drucken.

Bereits nach fünfzehn Druckzeilen mit dem frischen Farbband zeigte sich die Überlegenheit der Neukonstruktion des Farbbandantriebes. Mit einem fast unhörbaren Ratschen verabschiedete sich eine winzige Feder im Werte von vielleicht zwanzig Pfennig. Folge war, daß das Farbband nicht mehr mitlief, dadurch hämmerten die Nadeln immer kräftig auf die gleiche Stelle.

Natürlich war ich gerade mit der Kühlmittelbeschaffung beschäftigt, bemerkte den Fehler also relativ spät. Nun ist guter Rat teuer, gute Reparatur ist natürlich noch teurer. Aber man hat ja einen technischen Beruf gelernt und läßt sich nicht so leicht entmutigen. Also her mit der Werkzeugtasche und ohne Rücksicht auf die Garantie ran an den Feind, sprich Drucker. Der Fehler war auch schnell festgestellt, nämlich jene winzige Feder. Der leidgeprüfte Computerbesitzer ahnt schon, wie es weitergeht. Ein Ersatzteil muß her, und schon Edsel D. Murphy stellte in seinen Gesetzen über das Verhalten lebloser Gegenstände fest:

1. Die Ausfallhäufigkeit eines Bauteiles ist umgekehrt proportional zu seiner Wichtigkeit.
2. Die Zugänglichkeit eines Bauteiles ist umgekehrt propotional zu seiner Aufallhäufigkeit.
3. Die Erhältlichkeit eines Bauteiles ist umgekehrt proportional zu seiner Wichtigkeit und
4. was schiefgehen kann, geht auch schief.

Das Melodram mit dem sattsam bekannten Titel »Stadtrundfahrt« wurde also wieder einmal auf den Spielplan gesetzt, brachte aber diesmal ein völlig anderes Einspielergebnis: Nicht ein einziges Mal fiel die Bemerkung »Ham wir nich, sind aber bestellt...« Stattdessen Schulterzucken in den Kaufhäusern und Ratlosigkeit in den Fachgeschäften. »Bestellen wir, dauert aber mindestens sechs Wochen, wenn nich länger...«

Zum Überlaufen wurde das Faß dann von einem Servicetechniker gebracht (Firmenwerbung: »Eigener Service«): »Ich hab gerade eine Feder bestellt. Der Drucker da hinten hat den gleichen Fehler. Da muß ich wohl noch'n paar Federn nachbestellen...« — Der drohende Herzinfarkt ob solcher fachkundigen Äußerung muß schon deutlich zu erkennen gewesen sein, denn noch bevor ich die Gelegenheit fand, mich dazu zu äußern, wurde ich aus der »Werkstatt« herauskomplimentiert: »Keine Kunden in der Werkstatt...« Mein streßgeplagtes Ausse-

Fortsetzung auf Seite 22

Fortsetzung von Seite 20

hen muß dann bei meinen Arbeitskollegen Mitleid erweckt haben, denn nach etlichen Stunden der Suche in Grabbelkisten und Schubladen, fand sich das Material, aus dem man eine entpsrechende Feder herstellen konnte. Mit viel Mühe wurde eine Feder gebogen und eingebaut, schließlich kam doch noch das Erfolgserlebnis: Zehn Tage war der Drucker krank, jetzt druckt er wieder, Gottseidank... Mitnichten.

## Kein Frust

Durch das Drucken mit stehendem Farbband, war dieses doch arg in Mitleidenschaft gezogen worden. Es verhakte sich ständig, wodurch die Vorteile des Kassettensystems erst deutlich wurden. Statt mühsamer Fummelei genügt ein Griff und man hat die Kassette in der Hand und kann das Band wieder ordentlich einspulen. Wahrscheinlich hat man gerade dieses Vorteils wegen die Kassette eingeführt.

Es folgte der vorläufig letzte Akt des Dramas, eine Variation des Themas »Farbbandkauf«. Die Beschreibung des kurzen Einkaufsbummels erspare ich mir, irgendwie und irgendwann bekam ich schließlich eine neue Kassette. Hoffnungsfroh wurde selbige in den Drucker gelegt und dieser eingeschaltet. Bei dieser Gelegenheit stellte sich heraus, daß die Farbbandkassetten offenbar eine größere Odyssee hinter sich haben, wenn sie auf deutschen Ladentischen ankommen. Ein derart blasses Schriftbild hatte ich noch nicht einmal beim ersten Farbband, das bis zum letzten Tropfen im Drucker verblieben war...

Die Umtauschaktion gestaltete sich kurz und schmerzlos, schon nach fünfzehn Minuten intensiver Fachsimpelei war man bereit, zuzugeben, daß es möglicherweise tatsächlich etwas schwach war und händigte mir ein anderes Band aus.

Wer nun aus dem Berichteten den Schluß zieht, daß ich mit dem Drucker MPS 801 unzufrieden bin, der irrt. Kinderkrankheiten sind dazu da, überwunden zu werden, man kann keine Vorbeugungsmaßnahmen treffen, außer vielleicht schon mal etwas Baldrian für den Benutzer zu kaufen. Der MPS 801 bietet für sein Geld schon eine Menge Leistung, woran es mangelt, ist die Betreuung durch den sogenannten Fachhandel. Hier kann und muß noch eine Menge verbessert werden.

(Reinhard Schrutzki/gk)




**Es taucht sicherlich bei einigen der Wunsch auf, ihrem Commodore 64 das Sprechen beizubringen. Wir haben mit den SDP 120 eine hardwaremäßige Lösung getestet.**

Wie arbeitet nun dieser Sprachsynthesizer? Er benutzt das Verfahren der Phonemsynthese, wobei wir schon wieder bei der nächsten Frage sind: Was ist Phonemsynthese?

Mit diesem Verfahren ist es möglich, jedes beliebige Wort synthetisch wiederzugeben. »Phoneme« sind die vom Sprachsynthesizer erzeugten Laute, die sehr wenig Speicherplatz benötigen. Der SDP 120 benutzt 64 Phoneme (Einschließlich unhörbarer Steuercodes.) Daß es zwischen gesprochenen und geschriebenen Worten große Unterschiede gibt, merkt man spätestens wenn man versucht, einfache Worte in Phonemschreibweise zu erstellen.

Das Wort »Datei« wird aus den 5 Phonemen d - al - t - a - y zusammengesetzt. Bei »Diskette« wird die Schreibweise schon etwas schwieriger: d - il - ss - k - ael - t - ael. Am Titel dieser Zeitschrift hat man allerdings schon ganz schön zu kauen. Für »64'er« benötigt man folgende 14 Phoneme: f - il - r -ul - n - d - ss - ael-ch - t - ss - il - g - ael - r.

Welche Vorteile hat nun diese Art der Spracherzeugung? Der wohl größte Vorteil liegt im minimalen Speicherbedarf eines Wortes. Da ein Phonem durch jeweils ein Byte dargestellt wird, benötigt man für das Wort »Datei« nur 5 Byte. Ein weiterer Vorteil liegt darin, daß der Sprachsynthesizer jedes beliebige Wort oder jeden beliebigen Satz sagen kann, da er ja nicht an einen bestimmten Wortschatz gebunden ist. Das Phonem-Verfahren bringt jedoch nicht nur Vorteile, sondern birgt auch einige Nachteile. Vokale und einige Konsonanten können von Sprachsynthesizer recht gut nach-

# Hallo, hier spricht der Computer




geahmt werden, während einige synthetische Konsonanten kaum wiederzuerkennen sind. Darunter leidet natürlich die Verständlichkeit des Systems.

Vom Hersteller erhält man ein sorgfältig geschriebenes 28seitiges Handbuch, das neben technischen Einzelheiten und der Schaltungen einschließlich Platinenbestückungen, den Umgang mit dem Gerät erklärt und Hinweise gibt, wie man bei der Programmierung von Worten vorgehen soll. Es ist auch eine Liste von 250 aussprachemäßig definierten deutschen Wörtern beigefügt.

Das Herzstück des Sprachsynthesizers ist der Baustein SC-01A. Dieser Baustein besitzt einen Vorrat von 64 Lauten (=Phoneme), wovon drei Steuercodes sind (zwei Pausen und 1 Stop-Code). Die Phoneme können jeweils in vier sogenannten Betonungsstufen erzeugt werden, was bei den Steuerlauten natürlich bedeutungslos ist. Bei den Vokalen bewirkt das Setzen der Betonungsstufen ein starkes Anheben der Stimmlage, aber keine Erhöhung der Amplitude, was recht unnatürlich klingt. Bei den stimmhaften Konsonanten bewirkt das Setzen einer Betonung ebenfalls ein Anheben der Tonlage, was aber für die Betonung von Konsonanten im Deutschen ungeeignet ist. Eine Zunahme der Lautstärke erfolgt nicht, und so bleiben Akzente auf stimmlosen Konsonanten gänzlich ohne Wirkung. Im Deutschen werden aber Konsonanten ebenso wie die Vokale betont, was durch ein Anhebung der Lautstärke und gegebenfalls eine leicht Anhebung der Stimmlage erfolgen sollte. In vielen Fällen kann die Betonung von Silben durch Verlängerung der in ihnen enthaltenen Vokalen erzielt werden.

Mit den 64 Phonemen und den Betonungsstufen kommt man auf 4 mal 64 = 256 Codes, das entspricht genau einem Byte oder einem 8 Bit langen Datenwort. Dies ist für eine preisgünstige Konstruktion und für die einfache Programmierung vorteilhaft, vom Gesichtspunkt der Spracherzeugung her aber nur ein Minimum. Wünschenswert wäre eine mehrfach größere Anzahl von Grundlauten. Ähnlich verhält es sich bei den Betonungsstufen, die einen Kompromiß darstellen zwischen Vergrößerung der Amplitude (eigentliche Betonung) und der Anhebung der Stimmfrequenz. Da im Deutschen die resultierende Anhebung der Stimmfrequenz zu groß ist, führt dies zu unnatürlichen klingenden Anhebungen der Sprachmelodie. Besser wäre eine getrennte Steuerbarkeit von Amplitude und Stimmfrequenz.

Der Sprachsynthesizer wird an User-Port angeschloßen und kann somit von allen CBM-Geräten (CBM 2001 bis C 64) angesteuert werden. Das Design des Gerätes ist sehr einfach. An der Frontseite befinden sich der Ein/Aus-Schalter und zwei Drehregler um die Frequenz und Lautstärke zu ändern (siehe Bild).

Wie kann man nun die Sprachausgabe im eigenen Programm verwenden? Dies ist etwas aufwendig. Zuerst muß man mit Hilfe eines vom Hersteller mitgelieferten Programms den benötigten Wortschatz definieren. Dieser wird dann als sequentielles File auf Diskette gespeichert und kann vom eigenen Programm aus mit Hilfe eines zirka 40zeiligen Unterprogramms aufgerufen werden.

Der Sprachsynthesizer SDP 120 ist zum Preis von 487 Mark von Speech Design erhältlich.

**Fazit:**

Der Sprachsynthesizer SDP 120 findet sowohl als nette Spielerei für Hobbyprogrammier als auch in professionellen Programmen seine Anwendungsgebiete.

***Gesamtüberblick:***

| | |
|---|---|
| Verständlichkeit | : gut |
| Programmierung | : umständlich |
| Dokumentation | : sehr gut |
| Sprachbeispiele | : sehr gut |
| Gesamtnote | : gut-befriedigend |

(Christian Quirin Spitzner/rg)

Bezugadresse:
Speech Design, Altostraße 11, 8000 München 60

# GEHEIMNIS AUF

**Mit steigenden Umsatzzahlen des Commodore 64 gewann auch das dazugehörige Diskettenlaufwerk ständig an Popularität und verdrängte die Datasette. Wie es aufgebaut ist und welche Möglichkeiten die Hardware bietet, zeigt dieser Bericht.**

Zunächst ein paar Worte zur Geschichte: Entstanden ist das 1541-Laufwerk aus dem CBM 2031-Laufwerk, das bis auf die Schnittstelle identisch ist. Das 2031-Laufwerk arbeitet mit dem parallelen IEEE-488-Bus, während die 1541-Floppy Daten seriell überträgt. Früher hieß das Laufwerk allerdings noch 1540 und war zum Einsatz mit dem VC 20 vorgesehen. Mit Erscheinen des Commodore 64 erhielt es einen neuen Namen und durch ein neues Betriebssytem die Fähigkeit, sowohl mit dem VC 20 als auch mit dem C 64 zusammenarbeiten zu können. Die Änderung des Betriebssystems wurde wegen der unterschiedlichen Datenlesegeschwindigkeit der beiden Computer durchgeführt (tatsächlich ist der VC 20 schneller als der C 64). Meinen Erfahrungen nach funktioniert das 1541 am VC 20 aber auch ohne Umstellung (mit dem Befehl UI—) einwandfrei. Übrigens: Wer noch ein 1540-Laufwerk hat und dieses umrüsten möchte, kann dies durch Auslesen eines 1541-ROMs und Brennen auf EPROMs preisgünstig tun, ohne lange auf einen Umrüstsatz warten zu müssen.

Damit die gängigen EPROM-Typen (zum Beispiel der 2764) verwendet werden können, ist es allerdings notwendig, einige Pins umzuleiten. Ein solches Zwischenstück wird meines Wissens nach von Kalansky (siehe Expansions-Heft 7) geliefert.

## Einfaches Umrüsten des 1540

Die Erweiterung des Betriebssytems war allerdings nicht die einzige Veränderung, die seit dem Erscheinen des 1541 vorgenommen wurde. Die bis Weihnachten 1983 ausgelieferten 1541 funktionierten bei angeschlossenem 1526-Drucker beim Arbeiten mit relativen Dateien nicht einwandfrei, da diese Version eine Handshake-Leitung nicht überprüfte. Die Änderung des Betriebssystems des Druckers konnte daran natürlich nichts verbessern. Die nach Weihnachten 1983 verkauften Laufwerke haben meiner Erfahrung nach keinen Fehler mehr in ihrem DOS (Disk Operating System).

## Das 1541 ist eigentlich ein Computer

Damit habe ich schon auf eine Besonderheit des 1541-Laufwerks hingewiesen. Im Gegensatz zu vielen anderen Computersystemen ist das Commodore-Laufwerk ein sogenanntes »intelligentes« Peripheriegerät. Intelligent deswegen, weil im Laufwerk ein eigener Prozessor mit oben erwähntem eigenen Betriebssystem seine Arbeit leistet. Die VC 1541-Floppy wird deshalb mit Recht als eigener Computer bezeichnet. Neben dem 6502-Prozessor als CPU (Bild 1, Pfeil B) sorgen noch eine Reihe anderen Bausteine für den Datenaustausch mit dem Computer. Das sind im wesentlichen zwei 6522 Input/Output-Bausteine, von denen einer die Verwaltung des seriellen Busses übernimmt und der andere für die korrekte Motor-, Schreib- und Lesekopfsteuerung zuständig ist. Das 16 KByte große Betriebssytem

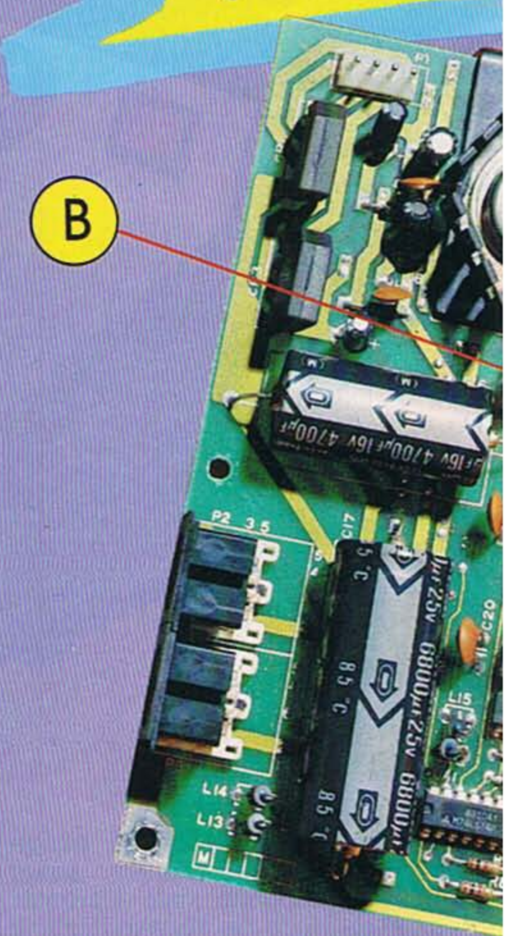
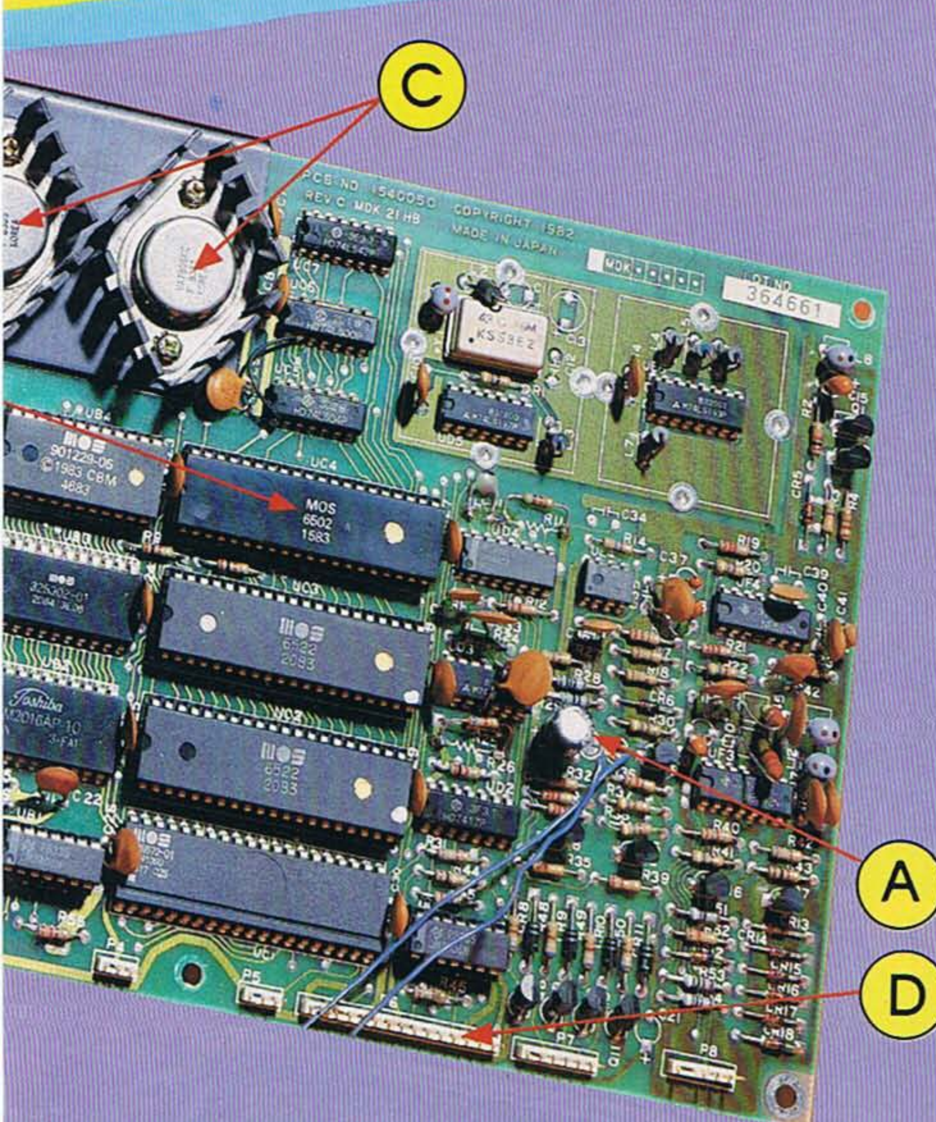



**Bild 1. Die DOS-Platine des 1541-Laufwerks**




ist in drei eigenen Bausteinen untergebracht. Zusätzlich stehen noch vier 2114-Bausteine mit insgesamt 2 KByte Pufferspeicher zur Verfügung.

Eigentlich eine ganze Menge, das für den relativ günstigen Preis geboten wird. Wie sieht es aber mit den Leistungen des Laufwerks im täglichen Betrieb aus?

## Igel oder Hase?

Wer von der Datasette auf die VC 1541-Floppy umsteigt, wird sicherlich zunächst von der Ladegeschwindigkeit, den Diskettenbefehlen und der Auswahlmöglichkeit des Directory begeistert sein. Wer aber, wie ich, schon mit anderen Computern und deren Laufwerken gearbeitet hat, ist von der relativ langsamen Datenübertragung enttäuscht. In Zahlen ausgedrückt, beträgt die Datenübertragungsgeschwindigkeit 0,4 KByte pro Sekunde (im Vergleich: Die CBM 2031 überträgt mit 1,8 KByte pro Sekunde). Außerdem ist die VC 1541-Floppy, jedenfalls die, die ich hatte, sehr störanfällig. Die bei längerem Betrieb (bei mir nach einem Monat) auftretenden Fehler machen sich zunächst in immer häufigeren Lesefehlern und Blinken der Kontrollampe bemerkbar. Später wird es dann unmöglich, noch einzelne Disketten zu lesen, außer sie wurden gerade formatiert. Dies sind typische Anzeichen einer Dejustierung des Schreib-/Lesekopfes.

Da diese Fehler bei meinem Laufwerk, auch nach Reparatur durch den Fachhandel, innerhalb kürzester Zeit wieder auftraten, war es an der Zeit, dem Problem auf die Spur zu kommen. Mit einem Disk-Monitor versuchte ich, bei meinem defekten Laufwerk eine frisch formatierte Diskette zu lesen. Das funktionierte auch auf allen Spuren einwandfrei, außer auf Spur 1. Die ausgegebene Fehlermeldung deutete auf eine fehlerhafte Formatierung hin. Da Spur 1 aber die äußerste auf der Diskette ist, schloß ich daraus, daß der Schreib-/Lesekopf die Position der Spur 1 nicht mehr exakt erreichen konnte. Mittlerweise war die Garantie auf mein Laufwerk abgelaufen (die erste Reparatur dauerte zwei Monate), deshalb entschloß ich mich, das Laufwerk zu öffnen.

## Abhilfe bei Funktionsstörungen

```
********************************
*    Achtung! Ziehen Sie bei    *
*  allen Arbeiten an offenen    *
*   elektrischen Geräten den    *
*          Netzstecker          *
********************************
```

Wer dies nachvollzieht, wird im Inneren die Platine mit dem Disk-Controller, dem Netzteil und dem eigentlichen Laufwerk (das von Alps, einem japanischen Elektrogiganten kommt) finden. Zusammengehalten werden die Einzelteile in einem Rahmen aus Preßblech. Nimmt man die Platine (bitte vorsichtig) ab, wird der Blick auf den Schreib-/Lesekopf (Bild 2, Pfeil F) und dessen Antrieb frei. Dieser Antrieb ist es aber, der meist für die Lesefehler verantwortlich ist. Die Funktionsweise ist

# GEHEIMNISSEN AUF DER SPUR

schnell erklärt: Ein Steppermotor überträgt über seine Achse (Bild 2, Pfeil E) die Bewegungsinformationen auf ein Rad, auf dem eine segmentförmige Erhebung ist. Um dieses Rad herum läuft ein Stahlband, das durch ein Zugrad auf Spannung gehalten wird. An dem Stahlband ist der Schreib-/Lesekopf befestigt und folgt somit den Bewegungen des Steppermotors. Beim sicherlich jedem Laufwerkbesitzer bekannten Klappern zu Beginn eines Formatiervorgangs justiert sich der Schreib-/Lesekopf, indem das auf dem Steppermotor angebrachte Rad mehrmals an einen Anschlag fährt. Dabei verschiebt sich im Laufe der Zeit das leider nur aufgesteckte Rad auf der Achse des Steppermotors. Daß der Schreib-/Lesekopf sich danach nicht mehr justieren kann, ist einleuchtend.

## Thermische Probleme

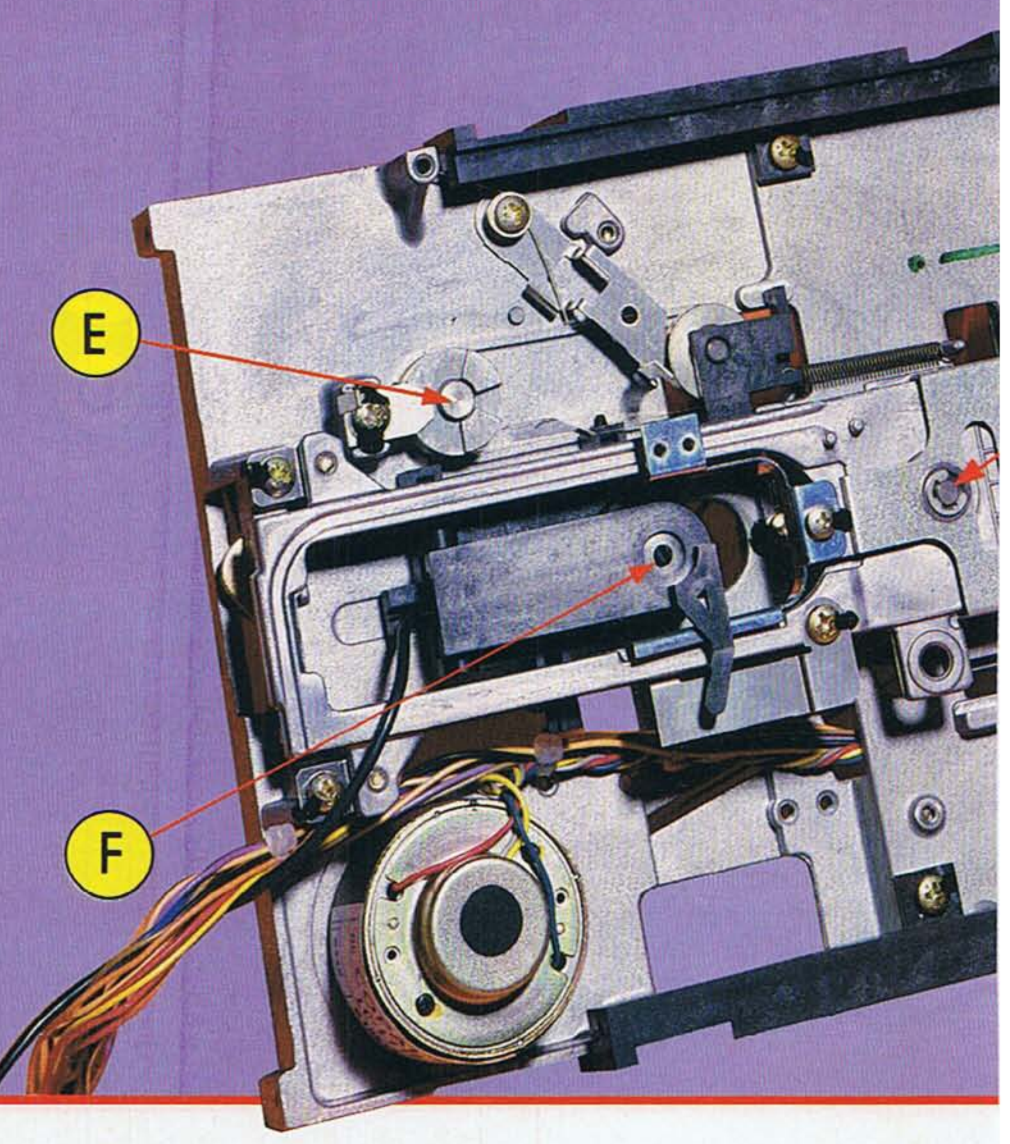
Verstärkt wird dieser Effekt noch durch die starke Erhitzung des Laufwerks bei längerem Betrieb. Diese ist zum einen darauf zurückzuführen, daß das Netzteil eingebaut ist, zum anderen entwickeln die beiden Spannungsregler (Bild 1, Pfeil C) Temperaturen über 50 Grad. Dies mag für die Elektronik ja noch verträglich sein, die Mechanik nimmt es übel. Da sich das Metall der Steppermotorachse und das des aufgesprengten Rades bei diesen Temperaturen unterschiedlich ausdehnt, wird das Rad auf der Achse leichter verdrehbar.

Welche Maßnahmen kann man aber treffen, um solche Verstellungen zu verhindern? Das Einfachste ist es, nach gründlicher Reinigung einen Tropfen Superkleber auf das Rad und die Achse zu geben. Dabei sollte man natürlich vorsichtig sein, damit nicht das Stahlbändchen verklebt. Eine andere, wesentlich elegantere Methode besteht darin, durch Achse und Rad ein Loch zu bohren und beide durch einen Splint zu sichern. Dazu ist natürlich die Demontage des Steppermotors notwendig. Deshalb sollte man diese Arbeiten nur dann vornehmen, wenn Kenntnis und einwandfreies Werkzeug vorhanden ist. Möglicherweise erklärt sich auch Ihre Fachwerkstätte, gegen ein gewisses Entgelt, dazu bereit. Dort könnte dann die selbstverständlich notwendige Justage des Schreib-/Lesekopfes vorgenommen werden. Sie können aber auch, mit etwas Geschick, die Justage mit einer auf einem einwandfreien Laufwerk formatierten (beziehungsweise einer Ihrer ersten Disketten) vornehmen. Die Einstellung mittels Disk-Monitor wird dann zwar nicht 100prozentig, aber ein störungsfreies Schreiben und Lesen ist auf jeden Fall zu erreichen.

Die zweite Fehlerursache beim 1541 liegt im Antrieb der Disketten (Bild 2, Pfeil G). Durch Verwendung von schmutzigen beziehungsweise alten Disketten geht oft die Haftung

## Schmutz als Fehlerursache

zwischen den Antriebsrädern und der Diskettenscheibe verloren. Die Folge ist eine zu geringe Umdrehungsgeschwindigkeit der Diskette (wenn sie sich überhaupt dreht), die meist mit wildem Blinken der Funktionsanzeige quittiert wird. Eine Ab-



hilfe schafft da nur die gründliche Reinigung des Andruckkopfes und des Antriebsrades mit Alkohol. Bei dieser Gelegenheit können Sie auch vorsichtig den Schreib-/Lesekopf mit einem Wattestäbchen säubern. Es ist empfehlenswert, die beiden Laufschienen des Schreib-/Lesekopfes ganz leicht (auch mit Wattestäbchen) einzuölen. Das Laufwerk wird dann leiser und zuverlässiger.

Sollten Ihnen die eben genannten Maßnahmen allerdings zu aufwendig vorkommen, ist es auf alle Fälle ratsam, für eine bessere Kühlung des Laufwerks zu sorgen. Dies ist entweder durch Einbau eines flachen Lüfters oder durch vollständiges Entfernen der Lüftungsschlitze im Gehäusedeckel möglich. Vermeiden Sie auf jeden Fall das Formatieren von Disketten, wenn das Laufwerk heiß ist.

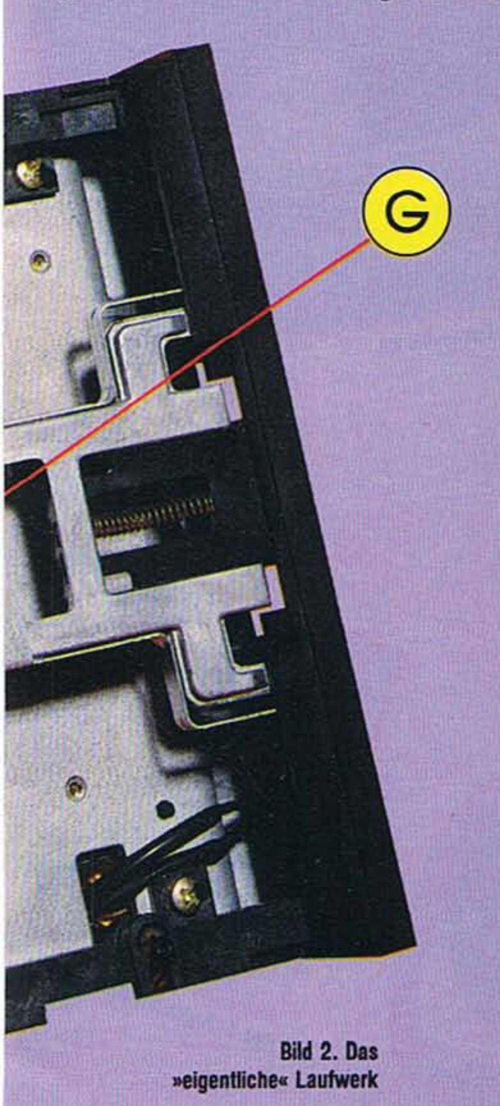



Bild 2. Das »eigentliche« Laufwerk

## Hardwareumstellung der Geräteadresse

Eine oft gestellte Frage ist auch die nach der hardwaremäßigen Umstellung der Geräteadresse. Die im Handbuch beschriebenen Teile sind nämlich auf der ganzen Platine nicht zu finden, weil das Floppy-Handbuch noch die alte, etwas größere und anders aufgebaute Platine beschreibt. Insgesamt gibt es drei verschiedene Platinen. In Bild 1 zeigt Pfeil A auf zwei Lötbrücken in der Form von zwei durch einen Steg verbundenen Halbkreisen. Durch Zertrennen der vorderen dieser beiden Lötbrücken läßt sich die Gerätenummer fest auf 9 einstellen. Ist es aber notwendig, das Laufwerk wechselweise mit 8 oder 9 zu betreiben, besteht die Möglichkeit, auf beide Halbkreise der vorderen Lötbrücke einen Draht zu führen und diesen mit einem Schalter zu verbinden. Nach Drücken des Schalters und Durchführen eines Resets stellt sich die neue Nummer automatisch ein. Der Schalter wird am besten an der Rückseite des Gehäusedeckels befestigt. Bei älteren (weißen) Diskettenlaufwerken befinden sich die beiden Lötbrücken übrigens nicht im vorderen Drittel der Platine, sondern, von vorne gesehen, hinten links. Auch hier wird wieder die vordere der beiden Leiterbahnen durchtrennt, um die Geräteadresse 9 einzustellen.

## Die Überbrückung des Schreibschutzes

Daß es beim 1541-Laufwerk möglich ist, auch einseitige Disketten (durch Herausschneiden eines zweiten Schreibschutzloches) und hardsektororientierte Disketten beidseitig zu verwenden, ist mittlerweile sicherlich bekannt. Daß es aber auch möglich ist, ohne diese Schnippelei Disketten auf der Rückseite zu bespielen, ist die nächste Erweiterung, die uns das 1541-Laufwerk bietet. Auch hier ist das Prinzip des Schreibschutzes schnell erklärt, denn eine einfache Lichtschranke prüft, ob sich eine Diskette mit oder ohne Schreibschutz im Laufwerk befindet. Dies hat mich vor langer Zeit mal eine Diskette gekostet, da ich aus Sparsamkeitsgründen durchsichtigen Tesafilm als Schreibschutz verwendet hatte.

Nun aber zum notwendigen Umbau. Auf der Platine befinden sich auf der linken Seite die Steckverbindungen für Laufwerkssteuerung, Datenübertragung und eine Leuchtdiode. Diese Steckplätze sind auf der Platine mit »p« und einer Nummer bezeichnet. Der für uns wichtige Steckplatz ist der mit der Bezeichnung P6. Die ersten beiden Kabel haben die Farben Orange und Lila. Lötet man zwischen diese beiden Kabel ebenfalls einen Schalter, so ist es möglich, Disketten von der Rückseite zu bespielen, ohne eine Einkerbung für den Schreibschutz machen zu müssen. Es ist empfehlenswert, einen Kippschalter zu verwenden, damit optisch erkennbar ist, ob der Schreibschutz aktiviert ist oder nicht. Zum Wechseln der Disketten ist der Schalter übrigens wieder in Normalstellung zu bringen, da bei geschlossenem Schreibschutzschalter ein Diskettenwechsel vom Laufwerk »nicht bemerkt« wird. Einige Nachteile dieser Arbeiten am Laufwerk sollten allerdings nicht verschwiegen werden: Durch Öffnen des Gerätes verlieren Sie jeden Garantieanspruch. Teile, die Sie bei Arbeiten an der 1541 zerstören (zum Beispiel durch statische Entladung oder zu heißen Lötkolben) werden Ihnen von keinem Händler ersetzt. Und achten Sie natürlich auf Ihre Sicherheit, indem Sie den Netzstecker ziehen und weglegen.

(Arnd Wängler)

Bild 8. Modell AK 300




**Das Angebot an Modems und Akustikkopplern wächst fast wöchentlich. Einen Überblick über die angebotenen Geräte soll unsere Marktübersicht, die keinen Anspruch auf Vollständigkeit erhebt, ermöglichen.**

**Automodem (Bild 1):**
Das Automodem ist ein Selbstbaumodem, das allerdings keine FTZ-Prüfnummer besitzt. Dieser Apparat wird zwischen Telefon und Akustikkoppler geschaltet. Es unterscheidet Computer von »normalen« Anrufern und kann auch als Wählautomat eingesetzt werden.

**Minimodem 3005 (Bild 2):**
Bei dem Minimodem fallen besonders die vier Leuchtdioden ins Auge. Neben der Diode, die die Betriebsbereitschaft anzeigt, und der Diode, die ein empfangenes Carrier-Signal bestätigt, sind die zwei anderen besonders interessant. Diese zeigen an, ob der Akustikkoppler Signale empfängt oder sendet. Durch diese Anzeigen wird ein Fehler in der Übertragung leichter lokalisierbar.

**Epson CX-21 (Bild 3):**
Der Epson Akustikkoppler ist wohl einer der verbreitesten Geräte. Er ist vor allem durch seine Zuverlässigkeit bekannt.

**Tandy (ohne Bild):**
Dieser Akustikkoppler ist der preiswerteste unserer Übersicht. Er soll nach der Meinung einiger Benutzer allerdings nicht die Zuverlässigkeit des Epson-Kopplers erreichen.

**CDI ACK 300 (Bild 4):**
Der ACK 300 sendet und empfängt mit 300 Baud im Voll-Duplex Betrieb.

**CDI ACK 1200 (wie Bild 4):**
Der ACK 1200 arbeitet mit 1200 Baud im Halb-Duplex Betrieb.

Bild 3. Epson CX-21

Bild 4. Modell ACK 300/300

Bild 1. Automodem




Bilder 6 und 7.
PR1

Bild 2. Mini-Modem 3005

**CDI AM 1200/300 (wie Bild 4):**
Dieses Gerät dient nur zum Senden. Die Übertragungsgeschwindigkeit kann zwischen 300 und 1200 Baud gewählt werden.

**CDI ACK 1200/75 (wie Bild 4):**
Dieser Koppler sendet mit 1200 Baud und empfängt mit 75 Baud im Voll-Duplex-Betrieb.

**CDI ACK 75/1200 (wie Bild 4):**
Dieser Akustikkoppler sendet mit 75 Baud und empfängt mit 1200 Baud im Voll-Duplex Betrieb. Als Besonderheit dieses Gerätes ist die BTX-Zulassung zu erwähnen.

**Comco CK 211 (ohne Bild):**
Dieser Akustikkoppler arbeitet mit 300 Baud im Halb- oder Voll-Duplex-Betrieb.

**Comco CK 311 (ohne Bild):**
Dieses Modell hat zusätzlich zu dem CK 211 noch den Originate/Answer Mode, besitzt aber keine FTZ-Nummer.

**Comco CK 2300 (ohne Bild):**
Bei diesem Gerät handelt es sich um ein tragbares Terminal. Integriert ist der CK 211.

**WS 2000 (Bild 5):**
Das Weltstandard 2300 Modem hat keine FTZ-Nummer. Es bietet vielfältige Möglichkeiten, die sich jeder »Hacker« wünschen würde. Doch durch das Veto der Bundespost dürfen diese nicht genutzt werden.

**PR1 (Bilder 6 und 7):**
Bei dem PR1 Akustikkoppler handelt es sich um ein Gerät, das mit Originate/Answer Mode arbeitet. Auch dieses Gerät besitzt keine FTZ-Nummer.

**AK 300 (Bild 8):**
Bei diesem Akustikkoppler handelt es sich um eines der neuesten Geräte mit FTZ-Nummer auf dem deutschen Markt. Der Koppler soll, nach Angaben der Hersteller, über dieselben Vorzüge wie der Epson CX-21 verfügen. Preislich liegt dieses Gerät jedoch einiges unter dem Epson-Koppler. Ein ausführlicher Testbericht wird sich in einem der nächsten Ausgaben anschließen.

**Das Bundespost-Modem (ohne Bild):**
Dieses Modem ist ausschließlich als Mietgerät über die Bundespost erhältlich.

Fortsetzung auf Seite 33

# Commodore 64

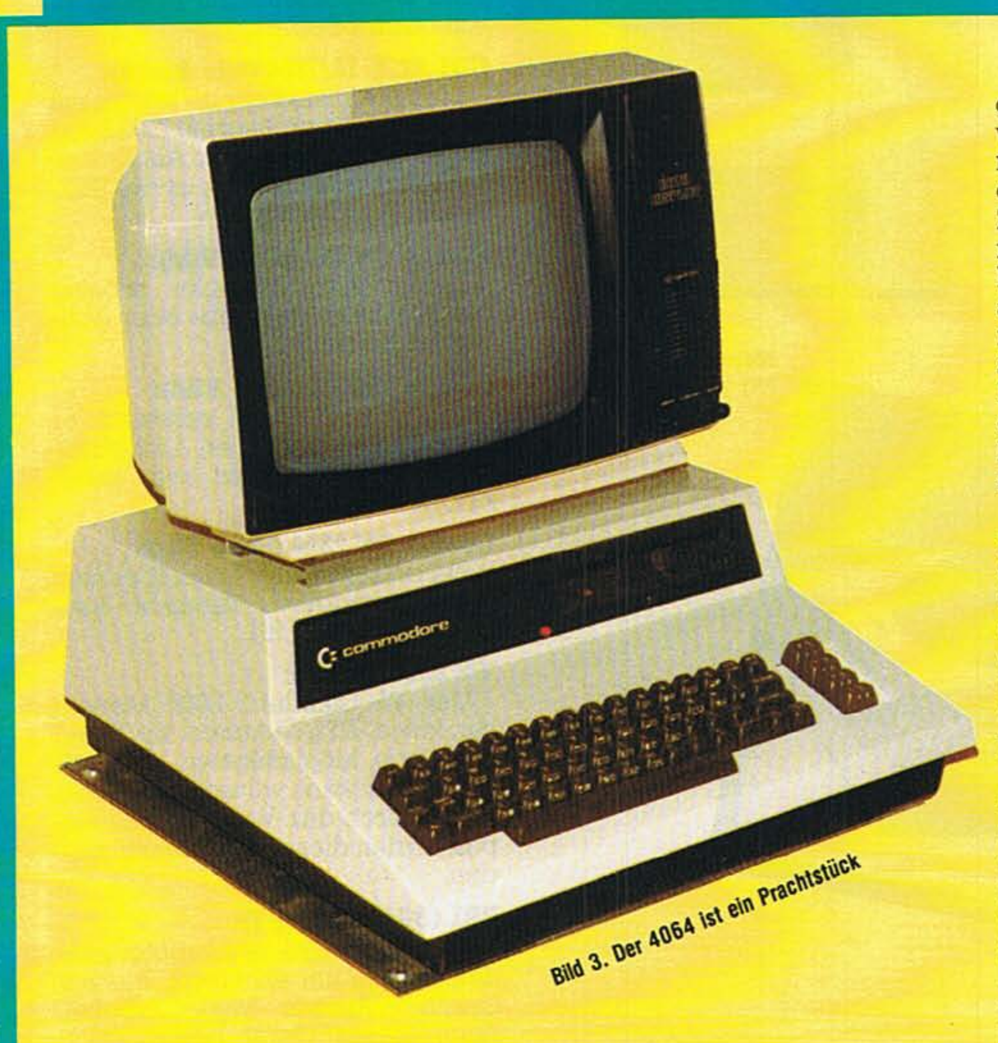


Bild 3. Der 4064 ist ein Prachtstück

Wenn Sie einen VC 20 oder einen C 64 besitzen, dann haben Sie sich bestimmt schon öfter Gedanken darüber gemacht, das äußere Erscheinungsbild Ihres Systems zu verbessern. Baut man die Geräte einmal so auf, wie vom Hersteller geplant und steckt diverse Erweiterungsplatinen dazu, dann entsteht auf dem Arbeitstisch ein recht unschöner Verhau von Leitungen. Abgesehen von dem dafür nötigen Platz, ein unbefriedigender Zustand.

Viele Verbesserungsvorschläge gehen dahin, nur den Computer auf dem Tisch zu belassen und den Rest so zu verstecken, daß er nicht stört. Den Weg, den ich zur Lösung des Problems beschritten habe, unterscheidet sich von diesen Vorschlägen wesentlich. Meine Absicht ist es, zwar alles auf dem Tisch zu lassen, dieses aber so kompakt aufzubauen, daß es kaum noch Platzprobleme gibt.

Die Idee dafür lieferten die »Urväter« der Büro-Computer, wie zum Beispiel der Pet oder seine Nachfolger. Warum sollte man den C 64 nicht in eines dieser Gehäuse einbauen? Eine praktische Lösung, die zudem noch einige weitere Vorteile

## Vom Pet gelernt

bietet: Auf einer Standfläche von zirka 50 mal 45 Zentimetern bringt man den Computer, den Bildschirm, das Floppy-Laufwerk und die Stromversorgung unter. Dabei ist immer noch genug Platz, um ein zweites Laufwerk und verschiedene Erweiterungen einzubauen. Soviel zum Platz, weitere Vorteile sind: Man hat nur noch ein Kabel zur Steckdose und eventuell noch eines zum Drucker. Zusätzlich macht das ganze dann noch einen wesentlich solideren Eindruck. Commodore selbst hat auch auf der Hannover Messe '83 einen 4064 für zirka 1500 Mark vorgestellt, allerdings ohne eingebaute Laufwerke und mit einem monochromen Bildschirm. Dieser findet hauptsächlich in einigen Schulen seine Anwendung.

Aber nun zur Praxis. Wie kann man sich diesen »Super C 64« bauen? Nun, einfach den Computer und das Laufwerk, mit allem was dazugehört, in ein Commodore-Gehäuse der 30/40XXer Serie einbauen, Bildschirm draufmontieren und fertig. Halt, ganz so einfach ist es auch wieder nicht, deshalb vorab eine Warnung: Wer eine Bohrmaschine für ein hochtechnisches Gerät und einen Lötkolben für eine südländische Frucht hält, sollte entweder die Finger davon lassen oder einen Bastler zu Rate ziehen. Aber auch für den professionellen Platinenkleber ist der Arbeitsaufwand nicht zu unterschätzen. Für den Prototyp habe ich sieben Tage gebraucht, mittlerweile schaffe ich es auch schon in zwei bis drei Tagen, (je nachdem, ob ich auf das Schlafen verzichte oder nicht).

Sollten Sie jetzt aber immer noch den notwendigen Mut zum Umbau haben, dann willkommen in der Welt der Commodore 4064-Besitzer. Dazu vorab eine Liste der unbedingt notwendigen Materialien und Werkzeuge.

Teileliste:

— Ein Gehäuse der CBM 30/40XXer Serie mit einer mindestens 5,5 cm hohen Frontplatte (sonst kann man meines Wissens kein Laufwerk einbauen).

— Einen Netztransformator mit vier Sekundärwicklungen (2x9 V und 2x15 V). Der Originaltrafo der 30/40XXer Serie ist sehr gut geeignet, man muß lediglich zwei Drähte,

# im neuen Kleid

**Ein altes Sprichwort sagt: »Kleider machen Leute«. In gewissem Maße gilt das auch für Computer. Ganz besonders aber für den Commodore 64, dessen Leistungsfähigkeit in keinem Verhältnis zu seinem äußeren Erscheinungsbild steht. Wie man diesem Mangel auf raffinierte Weise abhelfen kann, zeigt dieser Beitrag.**

die auf einen Pin gelötet sind, trennen. Selbstverständlich ist es möglich, einen Ringkerntrafo zu kaufen. Er erzeugt weniger magnetische Störungen und ist nur halb so groß. Dabei ist zu prüfen, ob die Versorgungsspannungen zusammengeführt werden dürfen, da Ringkerntrafos Sekundärwicklungen haben.

— Diverse Aluminium-Profilschienen
— Kühlprofil-Reste zur Kühlung der Computer- und Floppy-Platine
— Platinenstecker und -Stifte
— Eisenblech zur Abschirmung des Laufwerks
— Abstandbolzen mit Gewinde
— Jede Menge farbiger Drähte, Lüsterklemmen, Schrauben und Befestigungsteile
— Einen guten Lötkolben, eine Feile und mehrere Schraubenzieher
— Eine Bohrmaschine, möglichst auch mit Kreissägenvorsatz
— Einen Multitester.

Aus Bild 1 kann man entnehmen, wie ich die Platinen angeordnet habe. Man kann aber auch, wie Bild 2 zeigt, eine andere Plazierung der Platinen wählen.

Damit ist gleich ein wichtiger Punkt angesprochen. Beim Bau des 4064 muß man sich darüber im klaren sein, welche Erweiterungen später einmal angeschafft werden sollen. Dadurch erspart man sich den kompletten Umbau, wenn zum Beispiel ein zweites Laufwerk eingebaut werden soll. Aber auch Steckmodule brauchen Platz. Deshalb sollte der Expansions-Port sichtbar und zugänglich bleiben, eventuell bereitet man mit Flachbandkabel einen Modulsteckplatz am hinteren Ende des Gerätes vor.

Ich beabsichtige nicht, hier einen Bericht vorzulegen, aus dem jeder Handgriff mit Zeichnungen abzulesen ist. Dazu bräuchte ich wahrscheinlich das ganze 64'er Heft. Ich will nur die Reihenfolge der Arbeitsschritte aufzeigen und zu jedem Abschnitt einige grundsätzliche Hinweise geben, die vor Schäden und Zeitverlust bewahren. Denn die möglichen Fehler sind zahlreich und man muß ja nicht alle nachmachen.

Der erste Arbeitsschritt ist einfach — wenn er auch in der Seele schmerzt. Der C 64 und das Laufwerk werden ihres alten Kleides entblößt. Die Platine des C 64 wird einschließlich der Pappe aus dem Gehäuse herausgenommen. Ebenso das Laufwerk, bei dem zusätzlich noch das Metallgehäuse und der Netzfilter inklusive Sicherungsgehäuse ausgebaut werden. Sollten Sie dabei einen elektrischen Schlag bekommen haben, so rate ich Ihnen zwei Dinge:

1. Ziehen Sie den Netzstecker!
2. Bauen Sie Modellflugzeuge!

Haben Sie diese Hürde genommen, kann Ihnen sowieso alles egal sein, denn zusammen bekommen Sie die Originalgeräte auf keinen Fall mehr. Ich empfehle Ihnen folgende Arbeiten am Laufwerk: Ziehen Sie das schwarze Kabel des Lesekopfes hinten durch den Schlitz im Chassis und legen Sie alle anderen Kabel so, daß sie direkt unten heraushängen. Bereiten Sie ein Blechgehäuse mit den Außenmaßen des Floppys vor und vergessen Sie dabei nicht, für die kürzeste Verbindung der Kabel zur Platine ein gro-




Bild 4. Der Blick ins Innere des 4064

# Commodore 64 im neuen Kleid

ßes Loch in das Blech zu schneiden. Die Befestigungsschrauben im Chassis des 30/40XXer passen erstaunlich genau (zu amerikanischen Schrauben, deshalb mit M4 nachschneiden). Auch unter der Frontplatte ist alles für die Aufnahme zweier Laufwerke vorbereitet, die leider etwas kleiner gewesen sein mußten. Deshalb wird es notwendig, etwas zu feilen (zirka 2 Stunden). Die Befestigung des Laufwerks im Chassis ist am einfachsten mit vier Winkelblechen zu bewerkstelligen. Da man nicht gut bei geöffnetem Deckel mit Werkzeug hantieren kann, ist es zu empfehlen, das Laufwerk in geeigneter Schräglage in den Blechkasten zu montieren und den Kasten dann erst an den Deckel zu hängen. Wichtig ist dabei, das Laufwerkgehäuse nicht mechanisch zu belasten. Die Führung für die Diskette könnte sonst zu eng werden. Der Blechkasten darf also den schwarzen Kunststoff am Chassis nicht eng umschließen. Achten Sie

Hier sind Schlitze im CBM-Gehäuse
Floppy-Platine
Floppy-Platine
Rechner-Platine
vorne

Bild 1. Die erste Einbaumöglichkeit der Platine

auch darauf, daß keine beweglichen Teile irgendwo schleifen oder scheuern.

Jetzt aber zu den Platinen. Ich habe Wert darauf gelegt, an diesen nichts zu verändern. Deshalb wurden alle Anschlüsse über Stecker vorgenommen. Bei meiner Platinenanordnung mußte ich lediglich die seriellen Verbindungen mittels zweier Winkelstecker herstellen, da sonst das Gehäuse nicht mehr zugegangen wäre.

Der Zusammenbau geht relativ einfach und schnell. Die Hauptplatine wird mit den Abstandbolzen und der Pappe auf dem Boden des Chassis befestigt. Darüber kommt die Platine für das Laufwerk, das zusätzlich noch mit drei Schrauben an

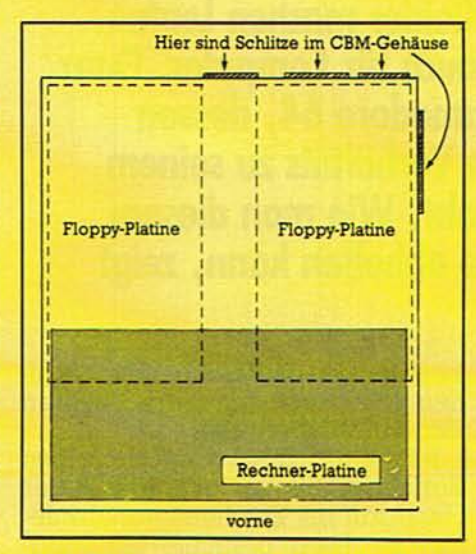



Bild 2. Die zweite Einbaumöglichkeit der Platine

der rechten Gehäuseseite auf einer Aluminiumschiene befestigt ist. Die andere Seite der Floppyplatine wird ebenfalls auf eine Aluminiumschiene geschraubt. Auf dieser findet später auch der Kühlkörper Platz, der unbedingt notwendig ist, denn es entstehen bei längerem Betrieb doch erhebliche Wärmemengen durch die beiden Regler.

Der Trafo wird am besten in der linken hinteren Ecke festgeschraubt, denn zum einen sind dort die Halterungen für den Originaltrafo und zum anderen ist dort noch am meisten Platz. Der Trafo wird grundsätzlich so montiert, daß die Sekundärklemmen sichtbar bleiben und das Hauptmagnetfeld nicht zum Monitor zeigt. Wer es ganz genau nimmt, kann den Trafo auch noch mit einem Blech abschirmen, unbedingt notwendig ist das aber nicht. Gut ist es, die Leitungen auf Schraubklemmen oder Lötleisten zu legen. Auch kann noch eine 220-Volt-Leitung für den Monitor und eventuell einen Lüftermotor mit vorbereitet werden.

Die Stromversorgung für den Computer (5 Volt) erhalten wir am einfachsten mit einem Festspannungsregler, der an die Gehäuserückseite geschraubt wird. Achtung! Unbedingt vorher ausmessen, denn eine zu hohe Spannung kann das vorzeitige Ende aller Bemühungen bedeuten. Beschriften Sie auch alle Leitungen und verdrillen Sie diese, damit es keine Verwechslungen und Störfelder gibt.

Ob Sie nun den Monitor nur auf das Gehäuse aufstellen oder fest verschrauben bleibt Ihnen überlassen. Bei einem festen Anbau ist aber darauf zu achten, daß dieser nicht endgültig ist, denn auch Monitore haben nur eine begrenzte Lebensdauer.

Eigentlich fehlen jetzt nur noch die Betriebsspannungslampen und die Resetleitung, die man sinnvollerweise auch gleich einbaut. Moment, ehe ich es vergesse: Die Tastatur haben Sie hoffentlich schon eingebaut. Dazu muß eine kleine Ecke vom Gehäuse vorsichtig weggefeilt werden. Jetzt haben Sie sich aber einen Probelauf verdient, vor dem Sie am besten nochmals alle Spannungen nachmessen und die Platinen von allem Staub und Spänen befreien.

Sollte alles funktionieren (hoffentlich), dann können Sie sich an die Gehäusekosmetik heranmachen. Dazu gehört das paßgenaue Feilen der Frontplatte und das Anbringen der besonders wichtigen Typenbezeichnung »4064«. Achten Sie in den ersten Wochen besonders auf die Temperaturentwicklung im Gehäuseinneren. Sollten Sie dabei zarte Rauchwolken aufsteigen sehen, empfehle ich Ihnen einen zusätzlichen Lüfter einzubauen.

Sollten Sie die Absicht haben, das Ganze auf sich zu nehmen, so kann ich, wenn nötig, CBM-Gehäuse mit Trafo in begrenzter Anzahl beschaffen. Es sollten aber auch bei verschiedenen Firmen, die die 30/40XXer-Platinen für andere Zwecke verwendet haben, noch solche Gehäuse erhältlich sein.

Die nächste Erweiterung meines 4064 ist eine Sicherung gegen Stromausfälle, so daß der Speicherinhalt etwa 10 bis 20 Minuten ohne Netz gehalten wird.

Übrigens ist es mir mit meinem 4064 mehrmals gelungen, selbst Besitzer größerer Computer, die dem Commodore 64 sonst nie einen Blick schenken würden, zu verblüffen: Sie hielten den 4064 für eine revolutionäre Neuentwicklung von Commodore. (H.Schröder/A.Wängler/gk)



# Modem + Akustikkoppler Marktübersicht

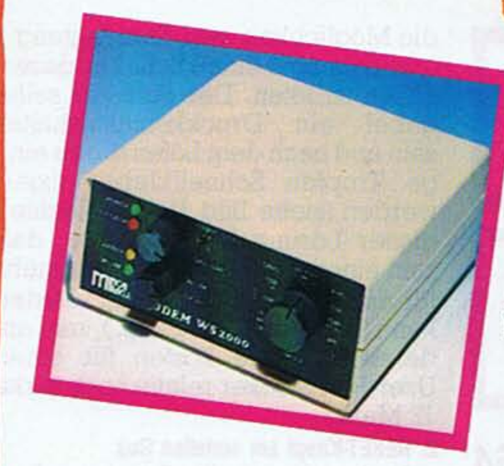



**Bild 5. Modell WS 2000**

Fortsetzung von Seite 29

**Das Bundespost-Einbaumodem (ohne Bild):**

Dieses Modem scheint eine sehr preisgünstige Alternative darzustellen. Mit einem Mietpreis von nur 19.90 Mark im Monat dürfte es manchen Konkurrenten aus dem Feld schlagen. Auch hier werden wir Hintergrundinformationen und einen Testbericht in einer der nächsten Ausgaben anschließen. (rg)

**Marktübersicht Modem und Akustikkoppler**

| Modell | FTZ | Preis | Anbieter |
|---|---|---|---|
| A-Tandy | ja | 398,— | Tandy Fachhandel |
| A-Epson | ja | 748,— | Epson Fachhandel |
| A-ACK300/300 | ja | 1137,72 | CDI Informationssysteme Tauentzienstr. 1 1000 Berlin 30 030/246015 |
| A-ACK1200/200 | ja | 1818,30 | CDI Informationssysteme siehe oben |
| A-ACK75/1200 | ja | 2046,30 | CDI Informationssysteme siehe oben |
| A-ACK1200/75 | ja | 2046,30 | CDI Informationssysteme siehe oben |
| AM-ACK1200/300 | ja | 444,60 | CDI Informationssysteme siehe oben |
| M-Automodem | nein | — | Selbstbau, Anleitung im nächsten Heft |
| M-WS2000 | nein | 700,— | Minor Miracles, GB Kontaktadresse: ACT Computer Schützenstr. 6 4000 Düsseldorf 1 0211/350188 |
| A-PR1 | nein | 550,— | K&N. Electronics Kontaktadresse ACT Computer siehe oben |
| A-AK300 | ja | 550,— | Software Express Hugo Vichoffstr. 84 4000 Düsseldorf 30 0211/424323 |
| M-Post Modem | ja | 80,— | Bundespost — Monatsmiete (bei 300 Baud) |
| M-Einbaumodem | ja | 19,90 | Bundespost — Monatsmiete |
| A-CK211 | ja | 1694,— | Comko Computersystem Gesellschaft m.b.H. Marsdorfer Str. 76 5000 Köln 40 0221/483051 |
| A-CK311 | nein | 2720,— | Comko Computersystem siehe oben |
| T-CK2300 | ja | 5525,— | Comko Computersystem siehe oben |
| Mini Modem 3005 | ja | 790,— | Modular Technology 4000 Düsseldorf 1 |

A = Akustikkoppler (Anschluß über den Telefonhörer)
M = Modem (Anschluß direkt ans Telefonnetz)
T = Terminal (Computer, Treibersoftware und Akustikkoppler in einer Einheit)
AM = Akustik-Muff (wie Akustikkoppler, aber nur zum Senden)

# RESET AM C 64

**Ohne Sicherheitskopie ist das Arbeiten mit dem C 64 eine riskante Sache. Zu oft stürzt ein Programm in der Entwicklungsphase ab. Also vor einem RUN immer abspeichern?**

Leider verzögert dieses häufige Abspeichern, besonders wenn nur mehrere kleine Änderungen vorgenommen werden müssen die Programmierarbeit sehr. Deshalb wäre es sicher wünschenswert am C 64 einen RESET-Knopf zu haben, wie er bei anderen Computern vorzufinden ist. Wichtig ist dabei natürlich, daß durch Betätigen dieses Knopfes der Speicherbereich nicht total gelöscht wird, sondern lediglich wieder für eine neue Befehlsaufnahme bereit ist.

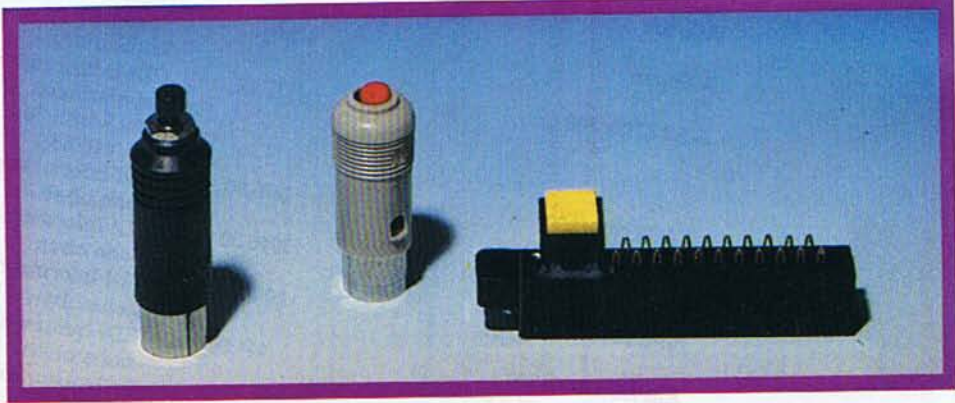


Bild 1. Verschiedene RESET-Schalter

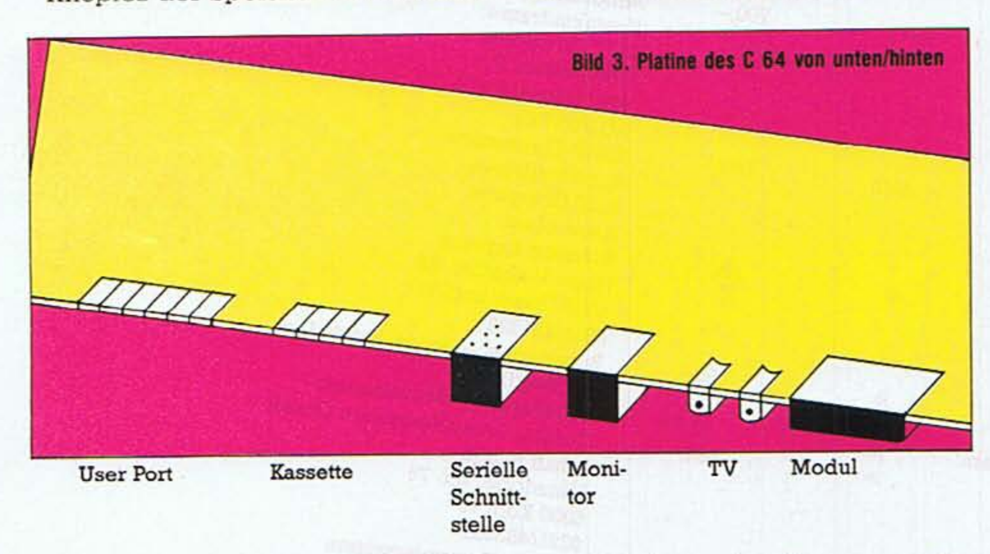



Bild 3. Platine des C 64 von unten/hinten

Da die beiden im Handbuch aufgeführten Möglichkeiten, den Computer wieder in einen aufnahmebereiten Zustand zu versetzen (RUN/STOP + RESTORE oder SYS 64738) oft zu keinem Resultat mehr führen, kann diesem Problem nur durch eine kleine Hardware-Erweiterung gelöst werden.

Das Prinzip dieses Schalters beruht auf einer Verbindung der RESET-Leitung des Prozessors (Pin 40) mit Masse (zum Beispiel Pin 21). Zur Realisierung dieser Verbindung bieten sich mehrere Ansatzpunkte an, die im folgenden beschrieben werden sollen:

**1. RESET-Knopf am User-Port**

Die an dieser Stelle nach außen geführte Platine des C 64 (User-Port) stellt an Leitung 1 GND (Masse) und an Leitung 3 RESET zur Verfügung. Durch Aufsetzen eines User-Port Steckers (zum Beispiel TRW 251-12-50-170 50-24SN-9) besteht nun die Möglichkeit, zwischen Leitung 1 und Leitung 3 einen Schalter dazwischen zu löten. Der Schalter sollte dabei ein Druckkontaktschalter sein und nach dem Löten durch einige Tropfen Schnellkleber fixiert werden (siehe Bild 1). Der Nachteil dieser Lösung besteht darin, daß zum einen der User-Port nicht mehr für andere Zwecke genutzt werden kann (zum Beispiel RS232..), zum anderen sind die Kosten für einen User-Port Stecker relativ hoch (zirka 12 Mark).

**2. RESET-Knopf am seriellen Bus**

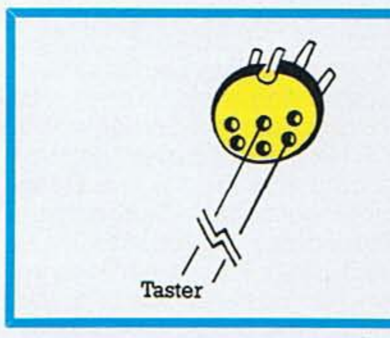
Auch der serielle Bus (zum Beispiel Buchse 2 am Floppy) bietet die beiden Leitungen GND und RESET (Bild 2). Diese befinden sich an den Stiften 2 (GND) und 6 (RESET). Als Stecker wird lediglich ein ganz gewöhnlicher sechspoliger DIN-Stekker, den es für wenige Mark in jedem Elektrogeschäft gibt, benötigt. Dort sollte man auch den weiterhin notwendigen Mini-Taster bekommen können. Dieser Mini-Taster wird nun auf die beiden Stifte 2 und 6 gelötet, so daß der Tastknopf oben aus dem Stecker herausragt (eventuell aufschneiden). Den ganzen Stecker braucht man dann nur noch auf den seriellen Bus aufstecken um einen preiswerten RESET-Knopf zu haben.

**3. RESET-Knopf fest eingebaut**

Die dritte Möglichkeit einen RESET-Knopf zu installieren, besteht darin, ihn fest in das Gehäuse des C 64 einzubauen. Dazu ist es notwendig, den Computer zu öffnen, die Tastatur auszustecken und die Platine auszubauen. Auf der Unterseite der Platine befinden sich die Lötkontakte des seriellen Busses. An dieser links und oben Mitte) über einen Druckschalter zu verbinden. Der links und oben mitte) über einen Druckschalter zu verbinden. Der Schalter selbst kann beispielsweise an der linken Geräteseite in einem gebohrten Loch Bild 4 befestigt werden.

Der Einbau des Schalters in den Computer ist allerdings dem Profi vorbehalten, da bei falschem Zusammenlöten der Kontakte, beziehungsweise bei zu starker Erhitzung der Platine, der Computer ernstlich beschädigt werden kann. Ferner ist zu bedenken, daß mit diesem Eingriff jeder Garantieanspruch verloren geht.

Hat man sich für einen der drei Wege entschieden und ihn realisiert, so wird man feststellen, daß nach dem kurzzeitigen Drücken der RESET-Taste das Programm verschwunden ist. Der Computer meldet sich nach einem RUN oder LIST Befehl mit einem freundlichen READY. Der Speicher scheint gelöscht zu sein. Das dies nicht so ist, sieht man durch den Einsatz eines Monitors. Betrachtet man sich den Basic-Programmstart, so wird man feststellen, daß das alte Programm noch vorhanden ist und nur die ersten drei Byte mit Nullen überschrieben wurden. Diese Nullen sagen dem Betriebssystem, daß hier das Programm zu Ende ist (genau wie nach einem NEW-Befehl). Um das Programm wieder lauffähig zu machen, müssen nur diese drei ersten Byte rekonstruiert werden.



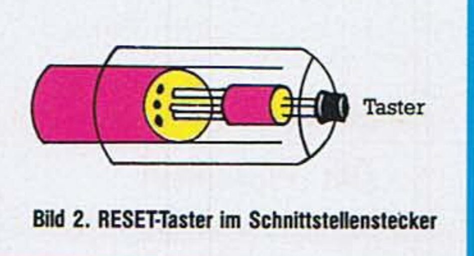




**Bild 2. RESET-Taster im Schnittstellenstecker**

Der Inhalt des ersten Byte ist immer Null, die beiden Folgebyte stellen einen Zeiger auf den Beginn der nächsten Basiczeile im Format Low-Byte, High-Byte dar. Diese beiden Byte werden mit der Adresse der nächsten Basiczeile gefüllt. Das Programm wird dann wieder lauffähig sein.

Die Rekonstruktion der Adresse der nächsten Basiczeile kann mit einem Monitor geschehen, indem das nächste Nullbyte im Speicher sucht und die darauf folgende Adresse in die beiden obigen Speicherzellen schreibt. Ein weitaus besseres Verfahren ist der Einsatz eines Programmes, das diese Arbeit für uns erledigt. Das beigefügte Programm OLD (siehe Listing 1) berechnet die gesuchte Adresse und schreibt sie in den Speicher. Damit wird der RESET beziehungsweise ein NEW aufgehoben und das Programm ist wieder vorhanden.

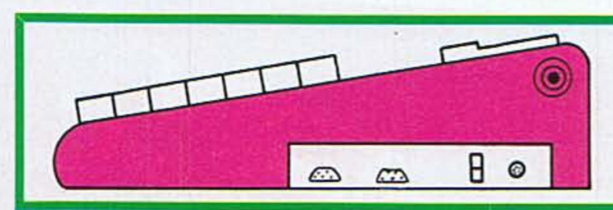


**Bild 4. RESET-Schalter im Gehäuse des C 64**

Es empfiehlt sich das Programm zunächst abzuspeichern und dann mit der Fehlersuche zu beginnen. Benötigt wird das OLD-Programm natürlich nur bei Basicprogrammen, ein vor dem RESET im Speicher befindliches Maschinenprogramm ist sofort wieder mit SYS startbar.

Das OLD-Programm selbst kann auf mehrere Arten geladen werden. Zum einen besteht die Möglichkeit, es mit einem Monitor im Kassettenpuffer (falls frei) abzulegen und als Maschinenprogramm abzuspeichern. Zum anderen kann es mit einem Basic-Lader (Listing 2) eingegeben werden. In diesem Fall wird das Programm ab $C000 abgelegt, da der Kassettenpuffer bei einem RESET ebenfalls gelöscht wird. Das Programm muß nun jeweils einmal nach Inbetriebnahme des Computers geladen und gestartet werden. Im Fall des Falles kann man es dann mit SYS 49152 aufrufen. Bei längeren Basicprogrammen sollte man es aber durch POKE 56,124 gegen überschreiben schützen.

Der Vorteil der ersten Möglichkeit besteht darin, daß man das OLD-Programm jederzeit laden und mit SYS 828 aufrufen kann, allerdings benötigt man einen Monitor. Die zweite Möglichkeit ist zwar einfach einzugeben, zerstört aber, hat man vergessen das Programm vorher zu laden, das zu rettende Basicprogramm.

```
.,   033c   a5 2b      lda $2b
.,   033e   a4 2c      ldy $2c
.,   0340   85 22      sta $22
.,   0342   84 23      sty $23
.,   0344   a0 03      ldy #$03
.,   0346   c8         iny
.,   0347   b1 22      lda ($22),y
.,   0349   d0 fb      bne $0346
.,   034b   c8         iny
.,   034c   98         tya
.,   034d   18         clc
.,   034e   65 22      adc $22
.,   0350   a0 00      ldy #$00
.,   0352   91 2b      sta ($2b),y
.,   0354   a5 23      lda $23
.,   0356   69 00      adc #$00
.,   0358   c8         iny
.,   0359   91 2b      sta ($2b),y
.,   035b   88         dey
.,   035c   a2 03      ldx #$03
.,   035e   e6 22      inc $22
.,   0360   d0 02      bne $0364
.,   0362   e6 23      inc $23
.,   0364   b1 22      lda ($22),y
.,   0366   d0 f4      bne $035c
.,   0368   ca         dex
.,   0369   d0 f3      bne $035e
.,   036b   a5 22      lda $22
.,   036d   69 02      adc #$02
.,   036f   85 2d      sta $2d
.,   0371   a5 23      lda $23
.,   0373   69 00      adc #$00
.,   0375   85 2e      sta $2e
.,   0377   60         rts
.
.
```

**Listing 1. Assemblerlisting des OLD-Programms**

Selbstverständlich arbeitet das OLD-Programm auch ohne die RESET-Taste, zum Beispiel nach einem irrtümlichen NEW-Befehl und stellt somit eine Basic-Erweiterung dar, die nicht nur in Verbindung mit der RESET - Taste ihre Anwendung findet.

(A. Wängler/rg)

```
0 data165,43,164,44,133,34,132,35,160,3,200,177,34,208,251,200,152,24,101
1 data34,160,0,145,43,165,35,105,0,200,145,43,136,162,3,230,34,208,2,230
2 data35,177,34,208,244,202,208,243,165,34,105,2,133,45,165,35,105,0,133
3 data46,96
10 forx=49152to49211:reada:pokex,a:next

ready.
```

**Listing 2. Basic-Lader des OLD-Programms**

# Programmodule selbst gemacht -Vergleic

Es kann durchaus sinnvoll sein, bestimmte Programme dauerhaft auf EPROMs zu speichern. Zur Programmierung der EPROMs ist es notwendig, ein sogenanntes EPROM-Programmiergerät, auch EPROM-Brenner genannt, einzusetzen.

Als Besitzer eines Commodore 64 hat man bereits einen Computer, der zur Steuerung der Programmierung eingesetzt werden kann. Alles was zusätzlich gebraucht wird, ist die entsprechende Treibersoftware und eine Platine, die für die notwendige Programmierspannung und die Aufnahme der EPROMs sorgt. Natürlich sollte es auch möglich sein, auf verschiedene EPROM-Typen umzuschalten. Alle diese Aufgaben übernehmen die zum Test angetretenen EPROM-Brenner.

## Die Testkandidaten

Das Feld der Testkandidaten bestand aus sieben Geräten. Zwei davon zum speziellen Einsatz am VC 20, zwei nur für den Commodore 64. Der Rest konnte an beiden Geräten wahlweise betrieben werden. Die Hersteller der beiden VC 20-Modelle waren Kalawsky und Weber Computertechnik. Die Modelle nur für den C 64 kamen von Kalawsky und von Bockstaller. Die übrigen Testteilnehmer kamen von Roßmüller und je zwei Geräte von Jeschke und Völzke. Wenn Sie nun mitgezählt haben, werden Sie feststellen, daß wir eigentlich neun Testkandidaten hatten. Da aber die Geräte von Jeschke und Völzke identisch sind, entfiel die Doppelzählung. Wichtigstes Bewertungskriterium war das Spektrum der programmierbaren EPROMs. Daneben wurden aber auch die Zuverlässigkeit, die Qualität und die Handhabung der Platine im Normalbetrieb bewertet. Zusätzlich flossen natürlich Sonderfunktionen wie zum Beispiel die Abschaltbarkeit des Brenners so oder ein Resetschalter mit in die Bewertung ein.

Bei der dazugehörigen Treibersoftware wurde vor allem auf den Nutzen und die Funktionalität der Programmierbefehle geachtet.

## Der Testablauf

Einem Modell einer futuristischen Stadt nicht unähnlich, präsentierte sich der EPROM-Brenner von Roßmüller. Zum Betrieb am User-Port vorgesehen ist die Platine (in tief dunklem Lötschutzlack) mit drei Schaltern und einer Resettaste ausgestattet. Der kleinste der Schalter dient der Umschaltung der Programmierspannung von 21 auf 25 Volt, damit auch ältere Typen des 2732 EPROMs »geschossen« werden können. Der größte der Schalter (sage und schreibe 5 cm hoch) dient der Umschaltung zwischen den verschiedenen EPROM-Typen.

## Einer für alle

Diese etwas unkonventionelle Lösung erwies sich im Test als sehr praktikabel, zumal die dazugehörende Software die Bedienung des Schalters unterstützt. Sehr angenehm fielen auch die Abschaltbarkeit der gesamten Platine (Schalter 3) und der durchgeführte User-Port auf. Das Programmiergerät braucht somit nicht entfernt zu werden, wenn es nicht benutzt wird. Der Aufnahmesockel für die EPROMs von Textool dürfte der wohl beste derzeit zur Verfügung stehende Sockel sein.

Gespannt waren wir, ob der positive Eindruck, den die Platine machte, auch bei der Treibersoftware erhalten bleiben würde. Keine Frage, wie sich schon bald herausstellte. Der Aufbau der Treibersoftware ist ansprechend in Form von Menüs gelöst worden. Das erste Menü erwartet die Eingabe des zu programmierenden EPROM-Typs (natürlich alles auf Tastendruck). Die dort aufgeführte Liste der programmierbaren EPROMs war beeindruckend und ließ keine Wünsche offen. Es können alle Typen von 1 KByte bis 32 KByte »geschossen« werden.

Nach der Wahl des EPROM-Typs erscheint das eigentliche Hauptmenü. Die »normalen« Befehle wie Programmieren, Vergleichen, Lesen, Leertest und Rückkehr in das Typenwahlmenü brauchen kaum erklärt werden, sie funktionieren logisch und einfach. Besonderes Augenmerk sollte aber den Befehlen »M« zum Verschieben von Speicherbereichen, »L« zum Laden von Kassette oder Diskette und »S« zum Speichern auf Kassette/Diskette geschenkt werden. Da der betreffende Speicherbereich dabei frei gewählt werden kann, stellen die Befehle eine sehr nützliche Hilfe dar.

Wünschenswert wäre allerdings noch die Möglichkeit, einen Hex-

**Was leisten die Geräte, mit denen der C 64 zum EPROM-Programmiergerät wird? Wo liegen die Leistungsunterschiede und welches Gerät bietet am meisten für seinen Preis? Unser Test gibt Antwort auf diese Fragen.**

# ...nstest Eprom-Brenner

| Hersteller / Merkmal | Roßmüller EPROM 64 | Völzke/Jeschke V 64 | Völzke/Jeschke Mod. 128 | Kalawsky EPROM-KING 64 | Kalawsky EPROM-KING 20 | Weber Computertechn. | Bockstaller |
|---|---|---|---|---|---|---|---|
| für Computertyp | C 64 | C 64/VC 20 | C 64/VC 20 | C 64 | VC 20 | VC 20 | C 64 |
| Anschluß an: | User-Port | User-Port | User-Port | Expans.Port | Exp.Port | Exp.Port | User-Port |
| Software auf | Kass./Disk | Kass./Disk | Kass./Disk | EPROM Kass./Disk | EPROM Kass./Disk | EPROM | Kass./Disk |
| Auch als Bausatz erhältl. | Ja | Nein | Nein | Nein | Nein | Nein | Nein |
| Eigenes Netzteil | Nein | Nein | Nein | Ja | Nein | Nein | Ja |
| Bus durchgeführt | Ja | Nein | Nein | Nein | Nein | Ja | Nein |
| Resetschalter | Ja | Nein | Nein | Ja | Ja | Nein | Ja |
| Programmierbare EPROM-Typen: 2508 | X | X | X | X | X | X | |
| 2516 | X | X | X | X | X | X | |
| 2716 | X | X | X | X | X | X | X |
| 2532 | X | X | X | X | X | X | X |
| 2732 | X | X | X | X | X | X | |
| 2764 | X | | X | X | X | X | |
| 27128 | X | | X | X | X | | |
| 27256 | X | | | | | | |
| Leertest | X | X | X | X | X | X | X |
| EPROM lesen | X | X | X | X | X | X | X |
| Programmieren | X | X | X | X | X | X | X |
| Auswahlpunkte der Treibersoftware: EPROM-Typ wählen | X | | | X | X | X | X |
| Speicherinh. verschieben | X | | | | | X | |
| Hexdump | | X | X | | | X | |
| Fehleranzeige | X | X | X | X | X | X | X |
| Speicher EPROM vergleichen | X | X | X | X | X | X | X |
| Repeatfunkt. | X | X | X | X | X | X | |
| Automatische Rechnerkennung | X | (X) | (X) | X | X | | |
| Preis: | Platine 99.– 199.– | 179.– | 249.– | 242.– | 242.– | 190.– | 270.– |

**Diese Tabelle zeigt die getesteten EPROM-Brenner in einer vergleichenden Übersicht**

Dump des betreffenden Speicherbereichs (zum Ansehen und Ändern) aufrufen zu können.

Der EPROM 64 von Roßmüller kommt ohne zusätzliches Netzteil aus. Im praktischen Betrieb konnten wir keinen Nachteil an dieser Lösung finden. Erhältlich ist der EPROM 64 für zirka 200 Mark als Fertiggerät und für zirka 100 Mark als Aufbauplatine für den Bastler.

## Das Schmuckstück

Silbrig glänzend, fast wie ein Schmuckstück, präsentierte sich der EPROM-King von Kalawsky. Der Realisierungsweg, der von dieser Firma beschritten wurde, unterscheidet sich von dem aller anderen Kandidaten. Der EPROM-King wird nicht am User-Port, sondern am Expansion-Port in Betrieb genommen. Damit ist der Vorteil verbunden, einem Modul mit Treibersoftware direkt in die auf der Platine vorhandene User-Port-Buchse einstecken zu können. Wird es aber vorgezogen, die Software von Kassette oder Diskette zu laden, kann hier ein Modul mit beliebiger Software eingesteckt werden. Bei der Stromversorgung findet sich die nächste Besonderheit dieses Geräts: Es wird von einem externen Netzteil mit der notwendigen Versorgungsspannung beliefert. Auf der Platine selber sind der (leider nicht optimale) Stecksockel, ein Resetschalter und eine Reihe von nützlichen LEDs angeordnet. Das falsche Einstecken des EPROMs wird durch die Lichtsignale der LEDs fast unmöglich gemacht.

Auch bei der Treibersoftware und der Anleitung wurde sehr darauf geachtet, jeden Anwenderfehler möglichst zu verhindern. Der sehr gute optische Eindruck der Software (viel Farbe, strukturierte Menüs) wurde sogar noch durch Tonsignale aufgewertet. Richtig durchgeführte Operationen werden durch einen hellen Ton, Fehler mit einem tiefen signalisiert. Auch beim EPROM-King ist das erste Menü der Wahl des EPROM-Typs vorbehalten. Hat man sich für einen der vielen Typen (1 KByte bis 16 KByte) entschieden, erscheint das Hauptmenü mit seinen umfangreichen Befehlen. Die Standardfunktionen sind natürlich auch hier vorhanden, werden aber durch Lade- und Speicherbefehle ergänzt. Das Einlesen des Directory ist dabei, ohne das Programm zu verlassen, möglich. Erfreulich ist der sinnvolle Einsatz der Funktionstasten. Schade, daß kein Maschinensprachemonitor eingebaut ist, denn sonst wäre die Treibersoftware als uneingeschränkt vorbildlich zu bezeichnen.

Der EPROM-King wird als Fertiggerät für zirka 245 Mark inklusive Netzteil und Software geliefert.

Die Version des EPROM-King für den VC 20 unterscheidet sich zwar äußerlich, von den gebotenen Leistungskriterien aber kaum, von der Version für den C 64. Lediglich auf das Netzteil wurde verzichtet und ein leider noch schlechterer Stecksockel für die EPROMs eingebaut. Die Treibersoftware ist für beide Geräte gleich komfortabel.

Nun waren schon zwei Kandidaten auf dem Weg zum Testsieg. Uns wurde langsam klar, daß es eine schwierige Entscheidung werden würde. Zuvor durften aber auch die restlichen Testkandidaten ihre Vorzüge beweisen. Ihnen voran die Mo-

Fortsetzung auf Seite 40

# Vergleichstest Eprom-Brenner

Fortsetzung von Seite 37

delle V 64 und Mod. 128 von Jeschke/Völzke. Alle Typen finden ihren Anschluß am User-Port und können sowohl am VC 20 als auch am C 64 eingesetzt werden. Ein Ladeprogramm sorgt dafür, daß die richtige Treibersoftware nachgeladen wird.

Auch hier findet wieder der exzellente Textoolsockel Verwendung. Zusätzlich sind, die bei diesen Modellen besonders wichtigen, Um-

## Gute Leistung, wenig Komfort

schalter auf der Platine angebracht. Mit Ihnen kann der richtige EPROM-Typ eingestellt werden. Hier liegt auch der Unterschied zwischen den Modellen V 64 und Mod. 128. Der EPROM-Brenner 128 kann alle EPROM-Typen neueren Datums (A, B, C) zwischen 1 KByte und 16 KByte programmieren. Beim Modell V 64 sind nur EPROMs bis 4 KByte möglich.

Da die Auswahl des EPROM-Typs bereits durch Einstellen der Schalter auf der Platine vorgenommen werden muß, entfallen diese Punkte im Menü der Treibersoftware. Dort finden sich aber zwei interessante Befehle: Erstens besteht die Möglichkeit, einen bestimmten Speicherbereich im hexadezimalen Format zu betrachten und zweites kann eine Funktion beliebig oft wiederholt werden. Da im Menü selber kein spezieller Punkt zum Laden oder Abspeichern von Programmen besteht, kann mit dem Hexdump-Befehl kontrolliert werden, ob das richtige Programm in den Speicher geladen wurde. »Fehlschüsse« von EPROMs sind deshalb bei sorgfältigem Arbeiten weniger wahrscheinlich. Verglichen mit den beiden Programmen der bisher getesteten Konkurrenten ist die Treibersoftware allerdings etwas einfach und optisch wenig ansprechend. Bei einem Preis von zirka 249 Mark für das große Modell und 179 für das kleine sind die Leistungen dieser beiden Geräte etwas mager ausgefallen.

Mit eigener Stromversorgung, aber ohne externes Netzteil? Kein Widerspruch, wenn man das Gerät von Bockstaller betrachtet. In einer Art Huckepackverfahren wurde die Platine mit einem eigenen Netzteil ausgestattet, das direkt hinter dem einfachen Stecksockel seinen Platz gefunden hat. Das Netzkabel muß natürlich trotzdem angeschlossen werden. Das bedeutet, daß (zwar mit Epoxyharz gesichert), 220 Volt auf der Platine sind. Ein unserer Meinung nach nicht sinnvolles Verfahren, zumal ein fester Sitz des relativ schweren Trafos zwar auf Monate hinaus garantiert werden kann, aber bei jahrelanger Beanspruchung... wer weiß? Die Leistungen dieses Gerätes halten sich, wie die Vergleichstabelle zeigt, in bescheidenen Grenzen.

## Der Selbstversorger

Auch die beigefügte Treibersoftware konnte weder von der Ausstattung noch der grafischen Darstellung mit den bisher getesteten konkurrieren. Die Auswahlpunkte des Menüs beschränkten sich auf die unverzichtbaren Standardfunktionen. Bei einem Preis von zirka 270 Mark kann diesem Gerät leider kein besonders gutes Preis/Leistungsverhältnis bescheinigt werden. Außerdem empfehlen wir, die Platine und die Anleitung (eine Seite!) nochmals zu überarbeiten.

## Basicprogramme einfach gebrannt

Das letzte zum Test angetretene Gerät von Weber Computertechnik ist zum Einsatz am VC 20 vorgesehen. Die Platine ist sehr übersichtlich aufgebaut und verfügt über den mittlerweile bekannten Textool Sockel. Der Expansion Port des VC 20 ist erfreulicherweise durchgeführt, so daß dieses zirka 190 Mark kostende Gerät im Normalbetrieb nicht entfernt zu werden braucht. Die eigentliche Besonderheit des EPROM-Brenners liegt in der Art seiner Treibersoftware. Sie ist in Form eines EPROMs auf der Platine eingesteckt und wird nach dem Anschalten lediglich über den SYS-Befehl aufgerufen. Das ist aber nicht der einzige Vorteil dieser Lösung: Es stehen zwei verschiedene Versionen von Software-EPROMs zur Verfügung. Mit dem Basic-EPROM ist es möglich, ein Basic-Programm in einen EPROM zu brennen. Das Maschinensprache-EPROM erfüllt die gleiche Aufgabe mit reinen Maschinensprache-Programmen.

Der Aufbau der Treibersoftware ist grafisch und inhaltlich sehr ansprechend und benutzerfreundlich gestaltet worden. Es wird oft verlangt, die gewünschte Funktion, falls sie einen Schaden verursachen könnte, nochmals zu bestätigen. Auch hier finden sich Befehle zum Verschieben von Speicherbereichen und eine Monitorfunktion. Die vorliegende Version dieses EPROM-Brenners kann mit der Basic-Software vier KByte große Programme auf EPROM brennen. In der Maschinensprachevers ion sind EPROMs von einem KByte bis acht KByte programmierbar. Ein ähnliches, mit erweiterten Funktionen (bis 32 KByte) ausgestattetes Programmiergerät für den C 64, lag zum Test leider noch nicht vor, soll aber mittlerweile erhältlich sein.

Die Entscheidung war nicht einfach. Zu verschieden sind die gebotenen Leistungen. An der Spitze des Testfeldes lagen, in einem Kopf-an-Kopf-Rennen, die beiden Geräte von Roßmüller und Kalawsky. Jeder mit seinen besonderen Vorzügen. Die Entscheidung fiel schließlich aufgrund der eingebauten Monitorfunktionen und der Fähigkeit, 32 KByte EPROMs zu programmieren, zugunsten des Modells von Roßmüller aus. Mit sehr guten Leistungen und komfortabler Treibersoftware Platz zwei für den Kalawsky EPROM-King. Leicht fiel die Entscheidung für den dritten Platz.

Obwohl die Software noch nicht voll befriedigen konnte, die Leistungen aber durchaus sehenswert sind, belegten die Modelle 128 von Jeschke/Völzke den dritten Platz. Bei den anderen Modellen wurde darauf verzichtet, eine weitere Platzierung vorzunehmen, weil jedes von Ihnen seine eigenen Vorteile besitzt, die bei der individuellen Kaufentscheidung mehr oder weniger ins Gewicht fallen. Bleibt zu hoffen, daß dieser Artikel wieder einige C 64/VC 20 Besitzer dazu angeregt hat, auch diese Anwendungsmöglichkeit ihres Computers verstärkt auszunutzen.

(Arnd Wängler/gk)




**Die Programmiersprache Comal für den C 64 soll eine Herausforderung an Pascal, Modula-2, PL/1 und C sein. Betrachtet man den Befehlsumfang und den Komfort dieser Sprache, so könnte dies zutreffen. Auch der Preis ist eine Sensation. Diese Sprache wird umsonst abgegeben.**

Viele C 64-Besitzer werden beim Programmieren festgestellt haben, daß das Commodore-Basic auf ihrem Computer nicht gerade ideal ist. In diesem Fall gibt es zwei Möglichkeiten. Entweder man kauft sich eine Basicerweiterung oder man sieht sich nach einer komfortableren Programmiersprache um. Inzwischen werden für den C 64 einige Sprachen angeboten. So auch Comal.

Doch wo liegen bei dieser Sprache die Vorteile und Unterschiede gegenüber Basic und den anderen Programmiersprachen? Zuerst wollen wir den Befehlsumfang von Comal betrachten.

Comal verfügt über 69 Befehle (Tabelle 1). Um die speziellen Möglichkeiten des C 64 zu nutzen, sind

| Comal Schlüsselbefehle | Bedeutung | Befehlsform |
|---|---|---|
| // | Kommentarzeile | //(Kommentar) |
| ABS | ergibt den Absolutwert einer Zahl | ABS(<numerischer Ausdruck>) |
| AND | logisches »und« | <Ausdruck> AND <Ausdruck> |
| APPEND | Start und Ende eines sequentiellen Files | OPEN(FILE)<filenum>, <filename>,APPEND |
| ATN | Arcustangens (rad) | ATN(<numerischer Ausdruck>) |
| AUTO | automatische Zeilennumerierung | (<Zeilenanfang>),(<Schrittweite>) |
| BASIC | zurück in Basic-Modus | BASIC |
| CASE | Mehrfachbedingung | CASE<Kontrollausdruck>(OF) |
| CAT | Directory | CAT(<Laufwerknummer>) |
| CHAIN | Laden und Starten von Diskettenprogrammen | CHAIN<filename> |
| CHR$ | Characterstring | CHR$(<numerischer Ausdruck>) |
| CLOSE | schließt Files | CLOSE((FILE)<filenummer>) |
| CLOSED | Geltungsbereich von Prozedur oder Funktionsvariablen begrenzen | PROC<procname>((Parameter)) (CLOSED)<br>FUNC<funcname>((Parameter)) (CLOSED) |
| CON | Programm fortsetzen | CON |
| COS | Cosinus (rad) | COS(<numerischer Ausdruck>) |
| DATA | Datazeilen | DATA<Zahl>,... |
| DEL | löscht Zeilen | DEL<Bereich> |
| DELETE | löscht File auf Disk | DELETE<filename> |
| DIM | Dimensionieren eines Feldes | DIM<string var>OF<max.char><br>DIM<string array>(<array index>) OF <max.char><br>DIM<arrayname>(<array index>) |
| DIV | Division | <Divident>DIV<Divisor> |
| DO | führe die folgenden Befehle aus | DO<Befehl> |
| EDIT | listet Programme ohne Zeileneinrückung | EDIT(<Bereich>) |
| ELIF | ELSE IF | ELIF<Ausdruck>THEN |
| ELSE | alternative Befehle in IF-Konstrukten | ELSE |
| END | beendet Programm | END |
| ENDCASE | beendet CASE-Konstrukt | ENDCASE |
| ENDFOR | beendet FOR-Konstrukt | ENDFOR |
| ENDFUNC | beendet Funktion | ENDFUNC(<function name>) |
| ENDIF | beendet IF-Konstrukt | ENDIF |
| ENDPROC | beendet Prozedur | ENDPROC(<procname>) |
| ENDWHILE | beendet WHILE-Konstrukt | ENDWHILE |
| ENTER | ein Programmsegment von Diskette »mergen« | ENTER <filename> |
| EOD | End Of Data flag | EOD |
| EOF | End Of File flag | EOF(<filenum>) |
| ESC | stop key pressed flag | ESC TRAP ESC<type> |
| EXEC | führe eine Prozedur aus | (EXEC)<procname>((<Übergabeparameter>)) |
| EXP | Exponent | EXP(<numerischer Ausdruck>) |
| FALSE | vordefinieren: Value = 0 | False |
| FILE | definiert welches File benutzt wird | INPUT FILE<filenum> (,<recnum>):<varlist><br>PRINT FILE<filenum> (,<recnum>):<varlist><br>READ FILE<filenum> (,<recnum>):<varlist><br>WRITE FILE<filenum> (,<recnum>):<varlist><br>OPEN (FILE)<filenum>, <filename>(,<type>)<br>CLOSE ((FILE)<filenum>) |
| FOR | beginnt FOR Schleife | FOR<var>=<start>TO<end> (STEP) (DO) |
| FUNC | beginnt Mehrzeilenfunktion | FUNK<name>((<params>)) (CLOSED) |
| GOTO | Sprungbefehl | GOTO<label name> |
| IF | Beginn IF-Konstrukt | IF<Bedingung>(THEN)<br>IF<bedingung>THEN<Befehl> |
| IN | lokalisiert String1 in String2 | <string1>IN<string2> |
| INPUT | Eingabe von Tastatur oder File | INPUT(<promt>:)<varLIST><br>INPUT FILE (s.o.) |
| INT | gibt die nächste ganze Zahl kleiner oder gleich | INT<numerischer Ausdruck> |
| KEY$ | prüft Tastatur | KEY$ |
| LABEL | vergibt Namen an Zeilen- | <label name> |

**Tabelle 1. Befehlsübersicht von Comal**




diese Befehle um drei weitere Befehlsblöcke erweitert worden. Diese Erweiterungen umfassen die Sprite, die HiRes- und die Turtle-Grafik sowie die Musik. Zwölf spezielle Befehle stehen für die Sprites zur Verfügung. Vergleicht man diese Erweiterung mit der Programmiersprache Logo, so stehen immer noch fünf zusätzliche Befehle zur Auswahl (Tabelle 1). Auch bei den HiRes- und Turtle-Grafik-Befehlen hat man gegenüber Logo fünf zusätzliche Befehle (Tabelle 2). Die Befehle sind sehr leicht zu handhaben und prägen sich schnell ein.

## Nun zu der Schnelligkeit

Comal ist mit einem »Three pass interpreter — run time compiler« ausgerüstet. Der erste »pass« setzt gleich beim Programmieren ein. Alle Zeilen werden, sobald sie abgeschlossen worden sind, auf die Syntax überprüft. Dies ist ein extremer Vorteil gegenüber Basic, denn 57 verschiedene Fehlermeldungen stehen Comal zur Verfügung. So ist ein Fehler leicht zu lokalisieren und eine langwierige Fehlersuche bleibt dem Programmierer erspart. Laut Hersteller sind Fehler drei bis zehn mal schneller zu beseitigen als in Basic.

Im zweiten »pass« werden die Befehlsstrukturen auf Korrektheit überprüft und die Sprungadressen in Absolutadressen umgerechnet. Dieser Durchlauf wird durch RUN ausgelöst und nimmt normalerweise weniger als eine Sekunde in Anspruch.

| Comal Schlüsselbefehle | Bedeutung | Befehlsform |
|---|---|---|
| LEN | definiert Länge eines Strings | LEN(<string Ausdr.>) Strings |
| LET | besetzt Variablen | := |
| LIST | listet Programm | LIST(<Bereich>)(<filename>) |
| LOAD | lädt Programm von Disk | Load<filename> |
| LOG | Logarithmus von n | LOG(<numerischer Ausdruck>) |
| MOD | ergibt REST der Division | <Divident>MOD<Divisor> |
| NEW | löscht Arbeitsspeicher | NEW |
| NOT | logisches NEIN | NOT<Bedingung> |
| NULL | bewirkt nichts | NULL |
| OF | Teil von DIM- und CASE-Konstrukten | CASE<Ausdruck>OF DIM<stringvar>OF<max.char> DIM<stringarray>(array index> OF<max.char> |
| OPEN | öffnet ein File | siehe »FILE« |
| OR | logisches ODER | <Bedingung>OR<Bedingung> |
| ORD | ergibt Zahl die einem Charakter entspricht | ORD(<string-Ausdruck>) |
| OTHERWISE | entspricht bei CASE dem ELSE bei IF | OTHERWISE <Bedingung> |
| OUTPUT | wählt Ausgabegerät | SELECT (OUTPUT)<type> |
| PASS | übergibt einen String an den Kommandokanal der Disk | PASS<disc command> |
| PEEK | ergibt den Inhalt einer Speicheradresse | PEEK<Speicheradresse> |
| POKE | besetzt Speicheradresse | POKE<Speicheradresse>, <Bedingung> |
| PRINT | schreibt auf Drucker, Bildschirm oder File | PRINT(FILE<filenum>:)(<items>) PRINT(FILE<filenum>:)USING <FORMAT>:<vars>(RANDOM file use:(file<filnum>,<recnum> :)) |
| PROC | startet Mehrzeilen Prozeduren | PROC<name>((<params>)) (CLOSED) |
| RANDOM | wahlfreier Zugriff auf Disk-File | OPEN FILE<filenum>,<filename), RANDOM<recln> |
| READ | Datas oder File lesen | READ<var list> READ FILE<filenum>(,<rec num>):<varlist> OPEN(FILE)<filenum>,<file name>,READ |
| REF | Parametervariable wird in einer Prozedur verwendet | REF<var> |
| RENUM | Renumber Programm | RENUM (<targetstart>)(, <Schrittweite>) |
| REPEAT | Beginn des REPEAT-Konstruktes | REPEAT |
| RESTORE | Datenzeiger auf ersten Data-Wert zurücksetzen | RESTORE |
| RND | Zufallszahl | RND(<num>) |
| RUN | Programmstart | RUN |
| SAVE | Programm auf Disk speichern | SAVE<filename> |
| SELECT | Ausgabegerät wählen | SELECT(OUTPUT)<type> |
| SGN | -1 wenn negativ<br>0 wenn 0<br>+1 wenn positiv | SGN(numerischer Ausdruck) |
| SIN | Sinus (rad) | SIN(numerischer Ausdruck) |
| SIZE | gibt benötigten Speicherplatz an | SIZE |
| SQR | Quadratwurzel | SQR(<numerischer Ausdruck>) |
| STATUS$ | Status vom Disk-Kanal | STATUS$ |
| STEP | Angabe für die Schrittweite in Schleifen | STEP<numerischer Ausdruck> |
| STOP | Ende des Programms | STOP |
| SYS | startet Maschienenprogramme | SYS(<Speicheradresse>) |
| TAB | Tabulator | TAB(<Spaltennummer>) |
| TAN | Tangens (rad) | TAN(<numerischer Ausdruck>) |
| THEN | Teil des IF-Konstruktes | THEN |
| TO | Endangabe für Schleifen | <startnum>TO<endnum> |
| TRAP | Ausschalten des Stopkey | TRAPESC<type> |
| TRUE | Variable mit eins vorbesetzen | TRUE |
| UNIT | Gerät definieren | OPEN FILE<#>,<nm>,UNIT<def> (,<sec>)(,<type>) |

**Tabelle 1. Befehlsübersicht von Comal (Fortsetzung)**

| | | |
|---|---|---|
| UNTIL | Ende der REPEAT-Schleife | UNTIL < Ausdruck > |
| USING | erlaubt formatierten Output | PRINT USING < format > : < varlist > |
| WHEN | Auswahl in CASE-Konstrukt | WHEN < list of values > |
| WHILE | Beginn WHILE-Konstrukt | WHILE < Ausdruck > (DO) (< Befehl >) |
| WRITE | Schreiben in ein File | WRITE FILE < filenum > (, < recnum >): < var list > OPEN (FILE) < filenum >, < file name >,WRITE |
| ZONE | Tabellenbereich | ZONE < tab interval > ZONE |
| **Spezielle Spritebefehle** | | |
| DATACOLLISION | Kollisionsabfrage mit Data | DATACOLLISION < sprite # >, < reset collisn flg? > |
| DEFINE | Sprite für späteren Gebrauch definieren | DEFINE < sprite definition num >, < 64 byte def$ > |
| HIDESPRITE | bestimmtes Sprite ausschalten | HIDESPRITE < sprite number > |
| IDENTIFY | einem Sprite eine Nummer zuordnen | IDENTIFY <sprite number >, < definition number > |
| (Bemerkung:Sprite 7 wird für TURTLE benutzt) | | |
| PRIORITY | gibt Sprite Priorität über Data | PRIOITY <sprite num >, <data priority? > |
| SPRITEBACK | setzt zwei Multicolor-Sprite Farben | SPRITEBACK < colorl >, <color2 > |
| SPRITECOLLISION | testet Spritekollision | SPRITECOLLISION < sprite # >, < reset collisn flg? > |
| SPRITECOLOR | setzt Sprite-Farbe | SPRITECOLOR < sprite number >, <color number > |
| SPRITEPOS | positioniert Sprite auf x,y Position | SPRITEPOS < sprite # >, < x coord >, <y coord > |
| SPRITESIZE | setzt Sprite-Größe | SPRITESIZE < sprite # >, <x expand? >, <y expand? > |


**Tabelle 2. TURTLE GRAFIK von Comal im Vergleich zu Logo**

| TURTLE GRAPHICS CHART Items | C 64 LOGO WORD | C 64 LOGO EXAMPLE | C 64 COMAL WORD | C 64 COMAL EXAMPLE |
|---|---|---|---|---|
| TURTLE CONTROL: | | | | |
| Move forward length | FORWARD | FORWARD 100 | FORWARD | FORWARD 100 |
| Move to a point | SETXY | SETXY 45 | SETXY | SETXY 4,5 |
| Move backward length | BACK | BACK THISFAR | BACK | BACK THISFAR |
| Home turtle | HOME | HOME | HOME | HOME |
| Turn turtle left | LEFT | LEFT 90 | LEFT | LEFT 90 |
| Turn turtle right | RIGHT | RIGHT MY-TURN | RIGHT | RIGHT MY-TURN |
| Turn to specific heading | SETHEADING | SETHEADING 0 | SETHEADING | SETHEADING 0 |
| Make turtle visible | SHOWTURTLE | SHOWTURTLE | SHOWTURTLE | SHOWTURTLE |
| Make turtle Invisible | HIDETURTLE | HIDETURTLE | HIDETURTLE | HIDETURTLE |
| Pen up off paper | PENUP | PENUP | PENUP | PENUP |
| Pen down on paper | PENDOWN | PENDOWN | PENDOWN | PENDOWN |
| Set pen color | PENCOLOR | PENCOLOR 4 | PENCOLOR | PENCOLOR 4 |
| Number of colors | 16 | | 16 | |
| Set size of turtle | — | — | TURTLESIZE | TURTLESIZE 6 |
| Plot a point | — | — | PLOT | PLOT 20,35 |
| Print text in graphics | ? | ? | PLOTTEXT | PLOTTEXT 1,1,»TEXT« |
| SCREEN AND COLOR CONTROL: | | | | |
| Clear graphics screen | CLEARSCREEN | CLEARSCREEN | CLEAR | CLEAR |
| Set to graphics mode | DRAW | DRAW | SETGRAPHIC | SETGRAPHIC |
| Set to text screen | NODRAW | NODRAW | SETTEXT | SETTEXT |
| Set background color | BACKGROUND | BACKGROUND 2 | BACKGROUND | BACKGROUND 2 |
| Set border color | — | — | BORDER | BORDER 4 |
| Fill in an area | — | — | FILL | FILL 25,20 |
| Full screen mode | FULLSCREEN | FULLSCREEN | FULLSCREEN | FULLSCREEN |
| Split screen mode | SPLITSCREEN | SPLITSCREEN | SPLITSCREEN | SPLITSCREEN |
| FUNCTION KEYS RESULTS: | | | | |
| F1 | TEXT SCREEN | | TEXT SCREEN | |
| F3 | SPLITSCREEN | | SPLITSCREEN | |
| F5 | FULLSCREEN | | FULLSCREEN | |
| F7 | GRAPHICS | | GRAPHICS | |




Der dritte »pass« ist der normale Programmlauf.

Im direkten Zeitvergleich ist Comal sechsmal schneller als Basic. Die Suche nach Strings soll nach Herstellerangabe sogar 79mal schneller als in Basic sein. Weiterhin soll man seine Programme drei- bis zehnmal schneller erstellen können. Auch ist zu erwähnen, daß Comal strukturiertes Programmieren voll unterstützt.

Besonders muß darauf hingewiesen werden, daß die Diskette mit dieser Programmiersprache nicht kopiergeschützt ist, und man sich so beliebig viele Backups herstellen kann.

## Diese Sprache gibt es fast kostenlos

Wie im Vorspann schon erwähnt, wird diese Sprache umsonst oder zum Selbstkostenpreis der Datenträger abgegeben. In der Anleitung selbst wird dazu aufgefordert, Comal zu kopieren und die Sprache dann an seine Freunde oder an seine Schule weiterzugeben. In Schleswig-Holstein ist Comal bereits an den Schulen eingeführt. Niedersachsens Schulen sollen bald folgen.

Comal ist über folgende Bezugsquelle zu erhalten:

Fa. INSTRUTEK
Christian Holmsgarde, 8700 Horsens, Dänemark

Ausdrücklich sei erwähnt, daß die 64'er Redaktion keine Kopien weitergeben kann. (rg)

Teil 2

# PASCAL

## leistungsfähiger und eleganter als Basic

**Funktionen, Prozeduren, dynamische Variablen sowie Mengenoperationen sind wichtige Elemente der Programmiersprache UCSD Pascal. Sie werden in diesem Bericht beschrieben und entsprechenden Basic-Lösungen gegenübergestellt. Abschließend werden sechs Pascal-Versionen verglichen.**

Listing 10 zeigt die Verwendung von Funktionen »Forward«-Deklaration, neue Statements und den Funktionsaufruf. Zuerst nun die Statements:

»DIV« und »MOD« dürften zum Teil sicher schon bekannt sein, dennoch zur Erinnerung: »MOD x« ergibt den Rest der Division einer Zahl durch »x«, »DIV x« hingegen den ganzzahligen Anteil des Quotienten.

Kurz: 345 MOD 7 = 2 und 345 DIV 7 = 49;

Mit der »Forward-Deklaration« existiert nun die notwendige Möglichkeit, eine Prozedur oder eine Funktion zu deklarieren ohne sie zu definieren. Stellen Sie sich vor, Sie hätten zwei Prozeduren P1 und P2. Sie beginnen mit der Deklaration der Prozedur P1 und bemerken, daß P1 intern P2 aufruft. Nun darf man bekanntlich nichts in einer Deklaration verwenden, was nicht schon vorher deklariert worden ist. Nun, werden Sie denken, das ist kein Problem, ich deklariere eben zuerst P2. Jetzt stellen Sie aber fest, daß P2 ihrerseits P1 aufruft. Was machen Sie nun? Für diesen Fall gibt es die »Forward«-Deklaration. Sie erlaubt es, wenigstens einmal den Namen der Prozedur oder Funktion sowie deren formale Parameter zu deklarieren, natürlich muß dann später die Definition nachgeholt werden, aber dann ohne die formalen Parameter. Wie das aussehen kann, steht im obigen Beispielprogramm.

Zur Funktion: Da die Funktion einen Wert ausgibt, kann man ihr auch einen Wert zuweisen. Das sieht man am Beginn des Hauptprogrammes. In der Funktion selbst muß man dem »Namen« der Funktion den Wert zuweisen, der in ihr bereitgestellt wurde. Damit hat nun die Funktion einen berechneten Übergabewert, mit welchem sie zurückkommt. Im Beispiel sind das die Zuweisungen: »QUERSUMME := x« und »QUERSUMME :=temp«.

```
PROGRAM QUERSUMMEN (INPUT,OUTPUT);
VAR        check,resultat : INTEGER;

 FUNCTION QUERSUMME (x :INTEGER) :INTEGER; FORWARD;

 PROCEDURE EINGABE;
 BEGIN
   PAGE (OUTPUT);
   WRITELN; WRITELN; WRITELN;
   WRITELN ('BITTE GEBEN SIE EINE GANZE ZAHL EIN,');
   WRITELN ('VON DER SIE DIE QUERSUMME WISSEN');
   WRITELN ('WOLLEN. MIT '0' KOENNEN SIE AUFHOEREN');
   WRITELN;
   READLN (check);
   resultat := QUERSUMME (check);
 END;          (* VON EINGABE *)

 FUNCTION QUERSUMME;
 VAR           temp : INTEGER;
 BEGIN
   x := ABS (x);  temp := 0;
   IF x < 10
     THEN QUERSUMME := x;
     ELSE REPEAT
            temp := x MOD 10 + temp;
            x := x DIV 10;
          UNTIL x <= 0;
   QUERSUMME := temp;
 END;          (* DER QUERSUMME *)

BEGIN          (* DES HAUPTPROGRAMMES *)
  QUERSUMME := 1;
  REPEAT
    EINGABE;
    WRITELN ('QUERSUMME: ',resultat);
  UNTIL QUERSUMME (check) = 0
END.           (* DES HAUPTPROGRAMMES *)
```

**Listing 10. Dank der Forward-Deklaration kann die Funktion Quersumme aufgerufen werden, ohne bereits deklariert zu sein.**

Mit der Funktion kann man sehr einfach rekursive Strukturen erschaffen. Eine rekursive Funktion ist einfach gesagt eine Funktion, die sich selbst aufruft, die sich selbst enthält. Man kann durch derartige Programmierung sehr viel an Speicherplatz sparen oder auch unvernünftig verschleudern, sie kann sehr schnell und effizient sein oder auch langsam und ineffizient. Rekursionen sind aber im allgemeinen »stackbelastend«. Wenn Sie sich das nächste Beispiel genau anschauen, werden Sie herausfinden weshalb. Die Funktion soll die sogenannten Fibboconacci-Zahlen ausgeben. Sie werden nach folgender Rekursionsformel berechnet:

$x(n+1) := x(n) + x(n-1)$ wobei $x(0) = x(1) = 1$ und $n => 1$

Im voraus sei gesagt, daß diese Funktion sehr ineffizient ist. Vielleicht merken Sie schon beim ersten Durchlesen (Listing 11) weshalb.

```
FUNCTION FIBBOCO (n :INTEGER) :INTEGER;
BEGIN
  IF n <= 1 THEN FIBBOCO := 1
            ELSE FIBBOCO := FIBBOCO (n-1)   +   FIBBOCO (n-2)
END;
```

Listing 11. Rekursive Funktion

Diese kleine rekursive Funktion berechnet also nach der Vorschrift ein beliebiges »x (n (>1))« bei Übergabe von »n«. Versuchen Sie doch einmal, den Verlauf für die Zahl »x (5)« nachzuvollziehen. Sie merken, daß sehr viele Aufrufe nötig sind, bis sich die Funktion nicht mehr selbst aufruft. Für den Fall, wo »n = 5« ist, sind 15 Aufrufe nötig. Das braucht Zeit und belastet eben den Stapel. Hier beträgt die maximale Rekursionstiefe 4. Ein weiteres schönes Beispiel für die Rekursion wäre die Lösung des Problems von Hanoi. Ich möchte sie hier nicht zeigen, der interessierte Leser findet sie in fast allen Büchern über strukturiertes Programmieren. Sie ist zwar sehr elegant, verbraucht aber sehr viel Speicherplatz und belastet den Stapel recht stark.

## Basic ohne Mengen

Eine weitere sehr mächtige Gruppe von Statements sind diejenigen, welche sich mit Mengen beschäftigen. Solche Befehle sind dem Basic gänzlich unbekannt.

Angenommen, Sie wollen in einem Programm nur eine bestimmte Eingabe erlauben, sagen wir alle Zahlen sowie die vier Operatoren »+,-,/,*«, so müssen Sie in Basic etwa das Programm in Listing 12 programmieren.

In der Schleife das Pascal-Programm wird getestet, ob die Eingabe der Bedingung genügt. Ist doch viel eleganter, nicht wahr? Nun bietet Pascal nicht nur solche einfachen Zugehörigkeitstests, sondern umfassende Mengenoperationen. Einige Beispiele dazu:

| TYPE | | |
|---|---|---|
| | farbe | = (ROT,BLAU, GRÜN,GELB, BRAUN,WEIß, SCHWARZ); |
| | tag | = (MON,DIE, MIT,DON,FRE-,SAM,SON); |
| | farbton | = SET OF farbe; |
| | besetzt | = SET OF tag; |
| | zahlen | = SET OF 1..500; |
| | zeichen | = SET OF char; |
| | buchstaben | = SET OF »A«..»Z«; |

Ein solches Set besteht jeweils aus der Potenzmenge aller zugehörigen Objekte. Sein Speicherbedarf kann unter Umständen immens sein. Man kann nun auf diesen Mengen alle Mengenoperationen durchführen: Vereinigungs-(+), Schnitt-(*) und Differenzmenge (-), Mengengleichheit (=) und -ungleichheit (< >, Teil- (<=) und Obermengen (=>) und Zugehörigkeiten (IN).

Sie sehen, man kann sehr universell mit Mengen hantieren (Listing 13). Es gibt dafür unzählige Anwendungsgebiete. Ihnen fallen sicher mehr ein. Ein kleiner Seitenblick auf Basic zeigt, daß solche Mengenoperationen dort überhaupt nicht unterstützt werden.

## Vielseitige Records

Ich komme nun zu einem Merkmal, das Wirth von Cobol übernommen hat, welcher aber typisch für Pascal ist. Es sind dies die Records. Schön, aber was ist denn ein Record?

Am besten ist es, wenn man einen Record mit einem Array vergleicht. Ein Array ist ja bekanntlich eine Anzahl Variablen, die denselben Namen (identifier) besitzen. Man unterscheidet die einzelnen Elemente dadurch, daß man sie indiziert. Aber einen großen Nachteil haben Arrays: Alle Elemente müssen von demselben Typ sein, das heißt man kann nicht in ein und demselben Array Strings, Reals und Booleans zugleich haben. Nun, das ist ja nicht weiter schlimm, man nimmt eben mehrere Arrays. Stellen Sie sich aber vor, Sie möchten eine einfache Adreßverwaltung herstellen und die einzelnen Elemente so einfach ansprechen wie die eines Arrays. Für jede Adresse brauchen Sie aber mehrere Felder, zum Beispiel für Namen, Wohnort, Postleitzahl etc. Wollen Sie noch irgendein Feld in der Adresse haben, welches sich für Berechnungen verwenden läßt, so haben Sie schon zwei Typen in der Adresse, nämlich Felder der Typen »ARRAY OF CHAR« oder, wenn vorhanden, »STRING« und, für Rechnungen, »REAL« oder »INTEGER«. Aufgrund dieser Betrachtung fällt

Listing 12. In Basic fehlen Mengen

```
10 GET a$: IF a$ = "" GOTO 10
20 IF a$ = "+" THEN  REM ADDITION
30 IF a$ = "-" THEN  REM SUBTRAKTION
40 IF a$ = "/" THEN  REM DIVISION
50 IF a$ = "*" THEN  REM MULTIPLIKATION
60 IF ASC(a$) < 48 OR ASC(a$) > 57 GOTO 10
70 REM  HIER FOLGT DAS WEITERE PROGRAMM

In Pascal ist das alles viel einfacher:

READ (c);   (* 'c' SEI VOM TYP 'CHAR' *)
WHILE c IN ('0'..'9','+','-','/','*') DO
  (* HIER FOLGT DAS WEITERE PROGRAMM *)
  (*                                  *)
```

```
PROGRAM MENGEN (INPUT,OUTPUT);
TYPE      farbe = (ROT,BLAU,GRUEN,GELB,BRAUN,WEISS,SCHWARZ);
            tag = (MON,DIE,MIT,DON,FRE,SAM,SON);
        farbton = SET OF farbe;
        besetzt = SET OF tag;
         zahlen = SET OF 1..500;
        zeichen = SET OF char;
     buchstaben = SET OF 'A'..'Z';
VAR       menge1,menge2 : farbton;
          woche1,woche2 : besetzt;
                antwort : besetzt;

BEGIN
  menge1 := (SCHWARZ,ROT,GRUEN,BLAU);
  menge2 := (SCHWARZ.GELB,BRAUN,BLAU,WEISS,ROT);
  woche1 := (MIT,FRE,SAM);
  woche2 := (MON,MIT,DON,FRE,SAM);
  WRITELN ('ZUERST DIE VEREINIGUNGSMENGE:');
  WRITELN (menge1 + menge2);
  WRITELN;
  WRITELN ('JETZT DIE SCHNITTMENGE:');
  WRITELN (menge1 * menge2);
  WRITELN;
  WRITELN ('NUN DIE DIFFERENZMENGE:');
  WRITELN (menge1 - menge2);
  WRITELN; WRITELN;
  WRITELN ('WANN KOENNEN SIE KOMMEN?');
  READLN (antwort);
  WRITELN;
  IF antwort <= woche2
     THEN WRITELN ('WIR HABEN IMMER ZEIT FUER SIE.');
  IF (antwort => woche2) AND ('DIE' IN antwort)
     THEN BEGIN
            WRITELN ('DIENSTAGS IST IMMER GESCHLOSSEN, ABER');
            WRITELN ('SONST KOENNEN SIE IMMER KOMMEN.');
          END;
  IF antwort => woche1
     THEN WRITELN ('SIE KOENNEN AUF JEDEN FALL IN DER 1.WOCHE KOMMEN.');
END.
```

Listing 13. Umgang mit Mengen, Aufzählungs- und Ausschnittyp

ein Array schon aus. Als »Ersatz« dafür bietet Pascal nun den Record. In ihm kann man nun beliebige Felder der verschiedensten Typen unterbringen. Die Records können nun so leicht angesprochen werden, wie die Elemente eines Arrays.

Ein Record wird im Typendeklarationsteil deklariert. Eine Vorbemerkung sei erlaubt: Alphanumerische Eingaben kann man im UCSD-Pascal als zum Typen »STRING« gehörig deklarieren, in Standardpascal muß man sie als »PACKED ARRAY OF CHAR« deklarieren. Beim Array kann man natürlich die Länge bestimmen, aber das ist bei Strings auch möglich und zwar folgendermaßen:

STRING (x) erzeugt Platz für einen String der Länge »x«. Ich habe im nächsten Beispiel (Listing 14) die Deklaration für eine Adreßdatei von 100 Adressen erstellt. Mit einem Index »i« habe ich nun Zugriff auf alle 100 Adressen.

Man könnte dies mit dem Statement »WITH« viel eleganter lösen, aber dieses Statement ist bis jetzt in keiner einzigen Pascal-Version für den Commodore 64 implementiert.

```
PROGRAM DATEI (INPUT,OUTPUT);
TYPE   maske = RECORD
                 VORNAME  : STRING[25];
                 NACHNAME : STRING[25];
                 STRASSE  : STRING[25];
                 NUMMER   : STRING[4];
                 PLZ      : STRING[6];
                 WOHNORT  : STRING[25];
                 TELEFON  : STRING[14];
                 SATZNR   : 0..100
               END;
       felder = (VORNAME,NACHNAME,STRASSE,NUMMER,PLZ,WOHNORT,TELEFON);
VAR    adresse : ARRAY [0..100] OF maske;
             i : felder;
       eingabe : STRING;
             c : CHAR;
             j : INTEGER;

 PROCEDURE BILDSCHIRM;
 BEGIN
   WRITELN ('BITTE');
   CASE i OF
     VORNAME : WRITE ('IHREN VORNAMEN :');
     NACHNAME: WRITE ('IHREN NACHNAMEN:');
     STRASSE : WRITE ('DIE STRASSE    :');
     NUMMER  : WRITE ('DIE HAUSNUMMER :');
     PLZ     : WRITE ('DIE POSTLEIZAHL:');
     WOHNORT : WRITE ('IHREN WOHNORT  :');
     TELEFON : WRITE ('IHRE TELEFONNUMMER:')
   END
 END;

 PROCEDURE ZUWEISUNG;
 BEGIN
   CASE i OF
     VORNAME : adresse[j].VORNAME  := eingabe;
     NACHNAME: adresse[j].NACHNAME:= eingabe;
     STRASSE : adresse[j].STRASSE  := eingabe;
     NUMMER  : adresse[j].NUMMER   := eingabe;
     PLZ     : adresse[j].PLZ      := eingabe;
     WOHNORT : adresse[j].WOHNORT  := eingabe;
     TELEFON : adresse[j].TELEFON  := eingabe
   END
 END;

BEGIN
   j := 0;
   REPEAT
     PAGE (OUTPUT);
     FOR i := VORNAME TO TELEFON DO
       BEGIN
         REPEAT
           BILDSCHIRM;
           READLN (eingabe);
           ZUWEISUNG;
           adresse[j].SATZNR := j;
           WRITELN ('SIND ALLE ANGABEN KORREKT?  J/N');
           READ (c);
         UNTIL c = 'J'
       END;
    WRITELN ('NOCH EINE EINGABE?');
    READ (c);
    j := j + 1;
  UNTIL (c <> 'J') OR (j > 100)
END.
```

**Listing 14. Ein Adreßsatz wird definiert und Adressen eingegeben**

Im Record sind deshalb nur Felder vom Typ »STRING« benützt worden, weil man mit ihnen leicht eine Eingaberoutine programmieren kann, wie ich es oben getan habe. Für verschiedene Typen, die natürlich ihrerseits auch wieder Records sein können, muß man mehrere »READLN«-Routinen für verschiedene Typen einführen oder gegebenenfalls eine »STRING to INTEGER«-Konversion durchführen.

```
PROGRAM DATEI (INPUT,OUTPUT,ADRESSE);
TYPE      maske = RECORD
                    (* SIHEHE OBEN *)
                  END;
          liste = FILE OF maske;
VAR    adresse : liste;
BEGIN
  REWRITE (adresse);
   (*                                                   *)
   (* DIESE STANDARDPROZEDUR ÖFFNET EIN FILE ZUM SCHREIBEN *)
   (* FALLS DIESES FILE SCHON EXISTIERT, WIRD ES GELÖSCHT  *)
   (*                                                   *)
 END.
```

**Listing 15. Eine Datei wird angelegt**

Mit dieser Eingaberoutine kann man nun alle 100 Adressen eingeben, wenn man will. Nun haben wir also schon eine einfache Adreßdatei aufgebaut. Das sollte man einmal in Basic versuchen, wobei das Programm aber vergleichbar viele Zeilen haben sollte.

Eine Routine zur Ausgabe der Adressen ist nach dem Studium des obigen Beispiels leicht erstellbar. Mit Angabe des entsprechenden Indexes hat man nun Zugriff auf alle Datensätze. Nun, was natürlich noch fehlt, ist die Abspeicherung der Datensätze auf irgendeinem externen Massenspeicher. Das ist in Pascal sehr einfach, ist aber mit zwei Einschränkungen behaftet. Die erste ist, daß man (in einfacher Weise) nur sequentiell abspeichern und lesen kann, die zweite, daß man ein File ausdrücklich als extern deklarieren muß, wenn man nicht will, daß es mit Beendigung des Programmes verschwindet (diese Tatsache kann man benützen, wenn man nur Zwischenwerte für zeitlich beschränkten Gebrauch extern abspeichern will). Die Deklaration und Eröffnung eines externen Files geschieht wie im Listing 15.

Hier gleich noch ein Wort zu »INPUT« und »OUTPUT«. Hier wird klar, was sie darstellen: Es sind Dateien. »INPUT« ist die Standard-Eingabedatei (meistens die Tastatur) und »OUTPUT« die Standard-Ausgabedatei (meistens der Bildschirm).

Anders als in Basic ist aber das Beschreiben des Files nicht so ohne

weiteres möglich. Jedes zu schreibende Element muß vorher in ein sogenanntes Dateifenster gebracht werden, denn nur dieses Fenster kann auf das File geschrieben werden. Das Dateifenster hat denselben Namen wie die dazugehörige Datei, man hängt ihm nur noch ein »^« hinten an. Die Standardprozedur, die nun das Dateifenster im File ablegt, heißt »PUT (dateifenstername)«. Das Fenster wird geschrieben und ist danach unbestimmt. Nachdem man also das Fenster belegt hat (hier also das Fenster »adresse^«), kann man es mit »PUT (adresse)« ins File schreiben. Aber Vorsicht: Man muß noch wissen, ob man an dieser Stelle überhaupt schreiben darf. Deshlab sollte man mit zum Beispiel »WHILE EOF (adresse) DO PUT (adresse)« jeweils testen, ob hier Platz frei ist oder nicht (EOF = End Of File). In unserem Beispiel (Listing 16) sähe dies so aus: Zuerst die Zuweisung (hier einmal mit »WITH«, vergleiche oben), dann das Abspeichern:

Damit hat man nun also einen Record abgespeichert. Das Zurückholen funktioniert ähnlich. Nehmen wir an, wir hätten mehrere Records abgespeichert, so befinden wir uns längst nicht mehr am Anfang des Files. Dorthin müssen wir aber, um erfolgreich einlesen zu können. Die Standardprozedur, die uns an den Anfang bringt und das File zum Lesen eröffnet, heißt »RESET (dateifenstername)«. Sie hat auch dafür gesorgt, daß bereits das erste Element des Files im Dateifenster »adresse^« bereitsteht. Analog zum Beispiel oben verhält sich die Belegung eines Records mit den Daten des Fensters. Dazu gibt es das Gegenstück zu »PUT«, nämlich die Standardprozedur »GET (dateifenstername)«. Damit man auch hier merkt, wann man aufhören soll zu lesen, sollte man mit dem Statement »WHILE NOT EOF (adresse) DO GET (adresse)« prüfen, ob das Ende des Files schon erreicht ist oder nicht. Sie sehen, mit dem Record läßt sich vieles sehr vereinfachen, wo es um große, verschiedenartige Datenmengen geht.

## Pascal: Auch dynamisch

Ziemlich am Anfang wurde festgestellt, daß Pascal eine statische Speicherverwaltung besitzt. Das stimmt nur bedingt. Es gibt in Pascal dynamische Speicherverwaltung, die sogar dynamischer ist, als diejenige in Basic. In Basic ist es bekanntermaßen nicht erlaubt, ein Feld mit »DIM« ein zweites Mal zu dimensionieren.

```
BEGIN
  WITH adresse^ DO
    BEGIN
      VORNAME   := adresse[j].VORNAME;
      NACHNAME  := adresse[j].NACHNAME;
      STRASSE   := adresse[j].STRASSE;
      NUMMER    := adresse[j].NUMMER;
      PLZ       := adresse[j].PLZ;
      WOHNORT   := adresse[j].WOHNORT;
      TELEFON   := adresse[j].TELEFON
    END;
  IF EOF THEN PUT (adresse)
         ELSE WRITELN ('BESETZT !');
END;
```

**Listing 16. Mit der With-Anweisung läßt sich die Dateneingabe in Listing 14 viel eleganter lösen**

In Pascal kann man dies erreichen, man muß nur anstelle der Arrays »Dynamische Variablen« verwenden. Man überläßt damit dem Computer die Verwaltung des Speicherplatzes jeder einzelnen Variablen, die dafür überall im Speicher verteilt sein können, zum Beispiel Variable »a« bei 3508, »b« bei 26876 etc. Anstelle von Indizes benützt man dann eben sogenannte »Zeiger« (pointers). Der Computer verwaltet alle diese Zeiger selbst, man kann die Werte dieser Zeiger somit niemals auslesen. Mit dieser Methode machen wir unsere Adreßverwaltung dynamisch. Das heißt man sollte problemlos löschen, sortieren, ändern und vor allem erweitern können. Es ist leicht ersichtlich, daß dies natürlich mit solchen Zeigern sehr einfach ist, weil man nicht die Datensätze verschieben, sondern nur die Zeiger ändern muß. Eine Variable wird zum Zeiger, indem man dem Typennamen, auf den er zeigen soll, ein »^« voranstellt. In unserem Beispiel:

```
TYPE  maske          = RECORD
                         (* SIEHE
                       OBEN *)
                       END;
      maskzeiger     =^maske;
VAR   adrl,adr2      : maskzeiger;
      j              : ^INTEGER;
```

Eine Deklaration schafft aber noch keinen Platz, da die Zeiger zuerst noch nicht definiert sind. Um nun wirklich im Speicher Platz für die dynamische Variable zu bekommen, gibt es die Standardprozedur »NEW (adrl) sucht Platz für den dynamischen Record »maske«, NEW (j) für die dynamische Variable »j«. Durch

adrl^VORNAME := »PETER«;

j^:= 9;

werden dem ersten Feld im dynamischen Record »adrl« der Name »PETER« zugewiesen und die dynamische Variable »j« bekommt via den Zeiger den Wert »9«. Beachten Sie den Unterschied:

**»j«** bezeichnet eine Zeigervariable auf ein Objekt vom Typ »INTEGER«.
**»j^«** bezeichnet eine dynamische Variable vom Typ »INTEGER«.
**»adrl«** bezeichnet eine Zeigervariable auf ein Objekt vom Typ »maske«.
**»adrl^«** bezeichnet eine ganze dynamische Variable vom Typ »maske«, das heißt einen ganzen dynamischen Record.
**»adrl^.feld«** bezeichnet ein Feld des dynamischen Records, auf den »adrl« zeigt.

Wenn man im Record ein Feld einführt, das selbst ein Zeiger ist, kann man sehr leicht verkettete Listen aufbauen etc. Speziell für diese Anwendung gibt es das Statement »NIL«. Wenn man nämlich einen Zeiger hat, der auf das Ende einer Liste zeigt, so darf er ja nichts mehr enthalten. Um diesen »Nichts«-Zustand genau zu definieren, kann man einem Zeiger den Wert »NIL« zuweisen. Die Verkettung beginnt mit einem Zeiger und endet mit »NIL«. Wenn man nun eine Sortierung der Datensätze durchführen will, so hat man im Wesentlichen »nur« die Zeiger neu zu berechnen, damit die Verkettung gemäß den Kriterien richtig verläuft. Es lassen sich so alle Arten von Datenstrukturen verwirklichen zum Beispiel Baum- oder Sternstrukturen.

Ich hoffe, daß man sich mit Hilfe meiner Ausführung einen kleinen Überblick über Pascal verschaffen konnte. Einen Schnellkurs über die Programmierung in Pascal ersetzt dieser Artikel sicher nicht. Der interessierte Leser wird ohnehin Fachliteratur lesen müssen, um alle Finessen von Pascal kennenzulernen.

## Sechs Pascal-Versionen im Vergleich

Mir sind bis jetzt sechs komplette Pascalsysteme bekannt: Pascal 64, Pascal 64 V3.0, beide von Data Becker, das Pascal von Abacus Software, G-Pascal, das KMMM-Pascal und das Oxford Pascal. Darüber hinaus gibt es noch einen 4-pass UCSD-Pascal-Compiler, über den ich aber noch nichts sagen kann. Die 35 reservierten Pascalstatements (AND, ARRAY, BEGIN, CASE, CONST, DIV, DO, DONWTO, ELSE, END, FILE, FOR ,FUNKTION, GOTO, IF, IN , LABEL, MOD, NIL, NOT, OF, OR, PACKED, PROCEDURE, PROGRAM, RECORD, REPEAT, SET, THEN, TO, TYPE, UNTIL, VAR, WHILE, WITH) werden nur von zwei Versionen vollständig unterstützt.

Im Pascal 64 fehlen: FILE nur bedingt, GOTO, IN, LABEL, NIL, PACKED, PROGRAM, RECORD, SET, TYPE und WITH.

Pascal 64 V3.0 enthält alle Standard-Pascalstatements!

Im Pascal von Abacus fehlen: FILE, GOTO, IN, LABEL, NIL, PACKED, PROGRAM, RECORD, SET und WITH.

Im G-Pascal fehlen: FILE, GOTO, IN, LABEL, NIL, PACKED, PROGRAM, RECORD, SET, TYPE und WITH.

Im KMMM-Pascal fehlen: GOTO, LABEL, SET und WITH.

Im Oxford-Pascal fehlt: kein einziges Standard-Pascalstatement!

Meiner Meinung nach ist ein Programm kein vernünftiges Pascal, wenn TYPE, SET und besonders RECORD fehlen, denn WHILE..DO, REPEAT..UNTIL, lokale Variabeln etc. bieten bekanntlich schon Simons Basic, welches dann auch noch Grafik und Sound recht gut unterstützt.

Es muß auch gesagt werden, daß lange nicht alle Standardprozeduren implementiert wurden. Ich möchte nun alle Versionen kurz besprechen.

## Pascal 64

Dieses Pascal wird in Basic emuliert! Es ist kein UCSD-Pascal, das heißt, komplette Stringbehandlung fehlt, wichtige Standardprozeduren wie NEW, EOF, PAGE, PUT, GET, PACK, UNPACK fehlen. Es besitzt dagegen eine kleine Grafikunterstützung, Sprites, zwei Rechengeschwindigkeiten (INTEGER/REAL), PEEK, POKE und SYS. Die Compilierzeit ist unerträglich lang. Nach Aussage eines Benutzers brauchte der Compiler für 234 Zeilen Sourcetext zirka 45 Minuten. Der Editor ist, weil es der eingebaute Full-Screen-Editor ist, natürlich zeilenorientiert und verlangt zum Teil unsichtbare GeSHIFTete SPACEs als Begrenzer von Zahlen, was man natürlich oft vergißt. Es werden darüber hinaus keine Editierhilfen geboten. Der Compiler ist nach dem Laden resident ab der Basiczeile 10000. Man kann daher den Pascalcompiler selbst nicht mit dem Austrocompiler oder Petspeed verschnellern, weil man die Zeile 1 bis 9999 für das eigene Programm benutzen muß. Fehlermeldungen nur wie in Basic (zum Beispiel ERROR in 20). Der P-Code ist nur zusammen mit einem Lader ausführbar. Das Handbuch hat einige Druckfehler und Ungenauigkeiten.

Alle reservierten Pascalstatements sowie alle Standardprozeduren sind implementiert! Weiterhin Mathematikfunktionen und Unterstützungen für die Programmierung von Hires- und Multicolor-Grafik und Sprites. Die Soundprogrammierung und der UCSD-Typ 'STRING', der aber im Typendeklarationsteil

## Pascal 64 V3.0

mit 'PACKED ARRAY (x..y) OF CHAR' definiert werden kann, werden nicht unterstützt. Bis zu vierdimensionale Arrays und komplexe Parameterübergabe an Prozeduren und Funktionen sind möglich. Als Anpassung an den Commodore 64 werden außerdem noch die relativen Files und ein bequemeres Diskettenhandling geboten. PEEK, POKE sowie Aufruf von Maschinenroutinen sind möglich. Eine Datenschnittstelle zum Profimat ist eingebaut. Als besonderes Feature gibt es die Interruptprocedure, die während eines jeden Interrupts ausgeführt wird, unabhängig davon, ob man sich noch im Pascalprogramm oder wieder im Basic befindet. Der Editor ist derselbe wie beim Pascal 64. Da aber der Compiler nicht schon wie beim Pascal 64 zur Erzeugung der Sourcefiles vorhanden sein muß, kann man Basiceditierhilfen wie zum Beispiel Exbasic Level II verwenden. Der langsame Compiler liest und compiliert von beziehungsweise auf Diskette. Nach erfolgreicher Compilierung muß ein spezielles Ladeprogramm eingeladen werden, das aus dem P-Code Maschinensprache und somit ein, ohne zusätzliche Hilfsprogramme, ablauffähiges Pascalprogramm erstellt. Erscheint ein Syntaxfehler, so steigt der Compiler aus ohne eine konkrete Angabe über Fehlerart und das Auftreten des Fehlers. Auch jetzt noch muß der Benutzer dem Compiler unverständlicherweise einige Parameter »von Hand« übergeben wie schon beim Pascal 64. Ein interessantes Detail für alle Beisitzer des alten Pascal 64: Nach Auskunft von Data Becker können alle diejenigen, die das alte Pascal 64 besitzen, dieses gegen zirka 50 Mark auf den Stand der neuen Pascalversion Pascal 64 V3. 0 aktualisieren lassen. Das neue Pascal 64 V3. 0 soll übrigens auch nur 99 Mark kosten.

## Pascal von Abacus

Dieses Pascal wird in Basic emuliert! Abcus-Pascal ist kein UCSD-Pascal: die komplette Stringbehandlung fehlt, wichtige Standardprozeduren wie NEW, EOF, PAGE, PUT, GET, PACK, UNPACK sowie Mathematikfunktionen fehlen ebenfalls. Es besteht aus drei Teilen: Editor (5,5 KByte), Compiler (10 KByte) und Interpreter (7,25 KByte). Der Editor arbeitet zeilenorientiert. Ein Pascalsourcefile wird als sequentielles File abgelegt, das heißt es ist möglich, einen anderen Editor zu benützen. Der Compiler bearbeitet zehn kurze Sourcezeilen in 25 Sekunden. Beim Auftreten eines Fehlers wird nur ein Fehlercode ausgegeben und der Compiler steigt aus. Man muß dann zum Beispiel den Editor selbst laden. Im Falle eines Fehlers wird man also in keiner Form unterstützt. Der P-Code läuft nur mit Interpreter.

## G-Pascal

G-Pascal ist kein UCSD-Pascal. Die komplette Stringbehandlung, wichtige Starndardprozeduren wie NEW, EOF, PAGE, PUT, GET, PACK, UNPACK, ORD sowie sämtliche Mathematikfunktionen, Type Real (kann nicht deklariert werden) und Boolean fehlen. Dafür bietet dieses Pascal eine extrem umfangreiche Sprite-, Grafik- und Soundunterstützung. Einiges mehr als zwei Dutzend Statements bietet G-Pascal allein für Grafik, Sound, Sprites (vor allem Kollisionen) und Timersteuerung. Es besitzt auch als einziges ein »GOTOXY«. Die Sprites scheinen IRQ-gesteuert. Maschinensprache-Routinen können aufgerufen werden. Abfragen für Joystick, Paddles, Lightpen sind vorhanden. Bitmanipulationen sind möglich. 16 KByte Maschinensprache werden geladen, und dann ist das Pascal inklusive Compiler und Editor resident. Der Editor ist ein zeilenorientierter Line-By-Line Editor recht beachtlichen Komforts, zum Beispiel mit Help-Befehl und Syntax-Checking Part vor der Compilierung. Fehlermeldungen werden im englischen Klartext ausgegeben. Der Compiler ist sehr schnell. Zudem bietet es einen Tracer, Debugger und Filer. Mit G-Pascal läßt sich bequem und übersichtlich arbeiten.

## KMMM-Pascal

KMMM-Pascal bietet komplette Stringbehandlungen sowie alle Mathematikfunktionen, Type Text und einen Zufallsgenerator. Von allen Standardprozeduren fehlen nur PACK, UNPACK, PAGE. Automati-

sche Typenkonversion (String → CHAR) und Bitmanipulationen sind möglich, wie auch Bitoperationen auf Integervariablen. Maschinensprache-Routinen können aufgerufen werden. PEEK und POKE sind auch möglich. Das Programm besteht aus drei Teilen: Editor (7,75 KByte), Editor/Compiler (23 KByte), Compiler (18,25 KByte) und Translator (19,5 KByte).

KMMM-Pascal erzeugt echten Maschinencode. Nach der Übersetzung kann man das Pascal-Programm abspeichern und nachher wie ein normales Maschinenprogramm laden. Es blendet das Basic-ROM aus, das heißt man hat neben dem Compiler noch runde 25 KBytes, für Pascalprogramme! Der Editor ist ein formatfreier Full Screen Editor überdurchschnittlichen Komforts (komfortabler als der eingebaute Editor), mit eingebautem Syntax Checking Part und belegten Funktionstasten. Eine häufig benötigte Steuercodesequenz kann als Macrobefehl definiert werden. Der Syntaxchecker spürt alle Syntaxfehler auf, wobei der Cursor die verursachende Stelle markiert und eine englische Klartextmeldung erscheint. Der Editor generiert ein sequentielles File, so daß man einen anderen Editor oder auch eine Textverarbeitung wie Easyscript benützen kann. Nach erfolgreichem Erstellen des Sourcefiles lädt man den

## Schnelle Compiler

Compiler, der bei Bedarf ein ASCII — oder »PETSCII«-Listing erstellt, während er (übrigens sehr schnell) compiliert. Der P-Code ist abspeicherbar. Nach der Compilierung wird der Translator nachgeladen, der nun den P-Code ziemlich schnell in Maschinencode übersetzt. Nach dessen Fertigstellung wird die Kontrolle wieder dem Basic übergeben. Die Zeiger sind automatisch so gesetzt worden, daß man das Programm sofort abspeichern kann. KMMM-Pascal unterstützt den IEEE-Bus und erlaubt die Verwendung von DOS 5.1. Im Lieferumfang gibt es auch noch mehr als zehn Demoprogramme sowie ein 70-seitiges Manual. Dieses Pascal kann gegen eine sehr geringe Gebühr dem jeweiligen aktuellen Stand angepaßt werden, was einem die Herstellerfirma garaniert. Der Hersteller gab auf eine entsprechende Anfrage bekannt, daß bis Ende 1984 das KMMM-Pascal den Sprachumfang eines Jensen/Wirth-Pascals besitzen soll, wodurch es dann ein sowohl im Handling als auch in bezug auf die Compilerleistung sehr überzeugendes Pascal sein wird. Von der Firma kann man auch gegen etwa 13 SFr. eine Programmbibliothek der schon bestehenden Programme erstehen. Diese Pascalversion wird von der Firma seit 1983 für den Commodore 64 angeboten und betreut. Alle Programme, außer denjenigen, die sich mit 'SET' oder 'WITH' befassen, wurden unter KMMM-Pascal erstellt.

## Oxford Pascal

Über dieses Pascal waren mir zu diesem Zeitpunkt erst diejenigen Informationen bekannt, die man einem englischen, nicht allzu umfangreichen Handbuch entnehmen kann. Es unterstützt alle 35 reservierten Pascalstatements und alle 40 Pascal-Standardprozeduren, alle Standardtypen (außer 'STRING', der aber leicht simuliert werden kann, siehe oben) sowie Mathematikfunktionen und Bitmanipulationen, auch auf Integern. Mengenoperationen sowie sogar variante Records(!) werden unterstützt. Darüber hinaus ist es auch an die Fähigkeiten des Commodore 64 angepaßt, was bedeutet, daß es Grafik und Sound unterstützt. Dazu besitzt es sehr komfortable Befehle, welche zum Beispiel in der Grafik Linien ziehen beziehungsweise löschen, Flächen ausfüllen beziehungsweise leeren und Punkte setzen beziehungsweise löschen oder prüfen können. Für den normalen Bildschirm gibt es die Prozedur »VDU«, die ein Zeichen an jeder beliebigen Stelle des Schirms ausgibt. Für den Sound gibt es die Prozedur »ENVEL«, die die Hüllkurve bestimmt, und für Frequenz, Wellenform und Dauer die Prozedur »VOICE« sowie eine eigene Prozedur »VOLUME«. Es unterstützt PEEK und POKE und erlaubt, in Pascal Maschinenprogramme zu editieren und als Pascalprozedur abzuspeicher. Weitere Features: Restore-Taste blockieren, Zugriff auf die Time Of Day-Clock, sedezimale Zahlenein- und -ausgabe, externe, linkbare Programm-Module, Forwarddeklarationen, Übergabeparameter dürfen auch Prozeduren und Funktionen sein, Prozeduren »PACK« und »UNPACK«, n-dimensionale Arrays. Es erzeugt Maschinencode. Es besitzt einen eigenen Editor, der im wesentlichen dem eingebauten Full-Screen-Editor entspricht, der aber um so nützliche Hilfen wie AUTO, DISK, NUMBER, FIND CHANGE, DELETE, PUT, GET, HEX, DECIMAL, DUMP, EX, LINK, COMPILE und einige Basicbefehle erweitert ist. Es gibt zwei Compiler. Der eine ist resident und compiliert und führt ein Programm sofort aus, was das Oxford Pascal vor allem für den Einsteiger, neben dem G-Pascal, sehr interessant macht. Der zweite Compiler ist ein Diskettencompiler, das heißt er liest und compiliert von beziehungsweise auf Diskette mit der Möglichkeit, externe Programm-Module einzubinden. Der Befehl »LOCATE« erzeugt danach ein ohne Hilfsmittel ablauffähiges Programm. Fehlermeldungen werden im englischen Klartext ausgegeben mit Angabe über die ungefähre Stelle im Listing.

## Nur drei Versionen brauchbar

Meine persönlichen Erfahrungen haben gezeigt, daß bis auf das KMMM-Pascal, das Oxford Pascal (soweit man ein Programm theoretisch beurteilen kann) und dann noch das Pascal 64 V.3 0 keine einzige der anderen Versionen befriedigen kann. So ist beispielsweise der erste Pascal 64 sein Geld nicht wert, das neue Pascal 64 V3.0 desselben hingegen ist in bezug auf die Leistung des Compilers das krasse Gegenteil vom Pascal 64. Einen komfortablen Editor besitzen beide Versionen nicht, was besonders beim neuen Pascal 64 V3.0 ein echtes Ärgernis darstellt. Die Fähigkeiten des neuen Compilers wurden gegenüber demjenigen des Pascal 64 drastisch verbesert. Wer sich ein zwar auch nicht überzeugendes aber dennoch in puncto Grafik und Geschwindigkeit dem Pascal 64 überlegenes Pascal zulegen möchte, der sollte sich das G-Pascal besorgen. Das Pascal von Abacus ist nur unwesentlich besser als das Pascal 64, denn es bietet eigentlich nur ein spezielles Pascalstatement 'TYPE'. Von allen Versionen ist wohl das G-Pascal das bequemste, weil es menügeesteuert ist. Das Pascal 64 ist bei Laufzeitfehlern bis auf das G-Pascal am leichtesten editierbar. Weil das G-Pascal und das KMMM-Pascal einen eingebauten Syntax Checking Part besitzen, sind sie bei Syntaxfehlern am leichtesten editierbar. Bei allen Versionen außer dem KMMM-Pascal, dem neuen Pascal 64 V3.0 und dem Oxford Pascal ist das Programm nur lauffähig,

Fortsetzung auf Seite 163

# NEUES VOM VC

Bildschirmformat:
25 Zeichen / 20 Zeilen

POKE 36864,8
POKE 36865,38
POKE 36867,40
POKE 36866,153

Bild 1. Bildschirmformat 25 x 20

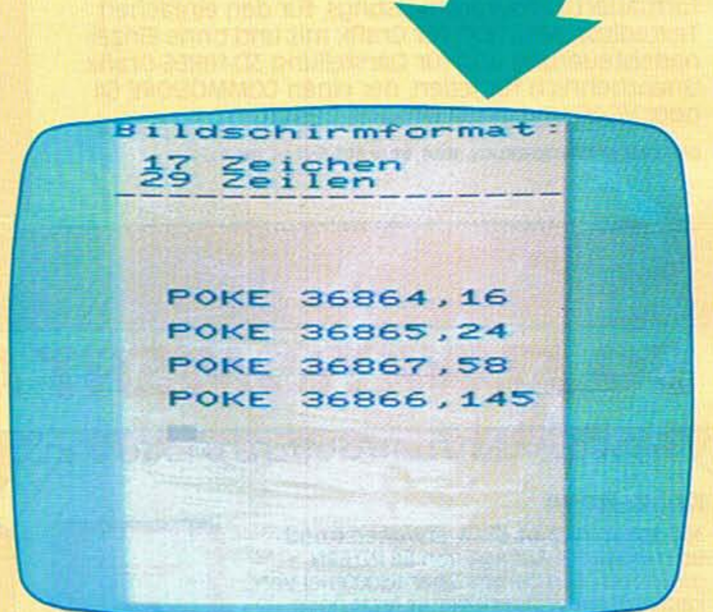


Bild 2. Bildschirmformat 17 x 29

**Der Video-Chip des VC 20 kann wesentlich mehr als nur die Bildschirmfarben steuern und Musik machen. So läßt sich zum Beispiel mit wenig Aufwand das Bildschirmformat ändern oder zwischen verschiedenen Bildschirmseiten umschalten. Auch die Erzeugung hochauflösender Grafik ist nicht so schwierig, wie es manchem Anfänger scheinen mag.**

Wie bei seinen »großen Brüdern« CBM 4032/8032 wird die Bildschirmdarstellung des VC 20 mit einem speziellen Videocontroller erzeugt. Das Betriebssystem stellt diesen Baustein nach dem Einschalten auf das bekannte Format von 22 Zeichen bei 23 Zeilen ein. Da diese Festlegung softwaremäßig erfolgt, ist es in Grenzen möglich, das Bildschirmformat nachträglich durch POKE-Befehle zu ändern. Leider werden solche interessanten Möglichkeiten des VC 20 (und C 64) von Commodore nur durch die Befehle PEEK und POKE unterstützt. Deshalb an dieser Stelle zunächst ein Exkurs zu den Bits und Bytes. Ein Byte setzt sich aus acht Bits zusammen, aber das wissen Sie ja sicherlich schon. Um in Basic einzelne Bits anzusprechen, muß eine Bit-Kombination zuerst in den entsprechenden Dezimalwert umgerechnet werden. Dies erreicht man mit der folgenden Beispielrechnung:

```
Bit:  7 6 5 4 3 2 1 0
Wert: 1 0 0 0 0 0 0 1
=1↑7*1+2↑6*0+2↑5*0+...+2↑0*1
oder: 2↑7+2↑0
```

Das Ergebnis aus 2↑7+2↑0 lautet 128+1 also 129. Mit den beiden Basic-Befehlen AND und OR können die Bits gezielt verändert werden. Hierzu ein Beispiel: »R=128 : V=1 : PRINT R OR V«. In Bit-Schreibweise sieht die Rechnung wie folgt aus:

```
R:1 0 0 0 0   0 0 0: 128
V:0 0 0 0 0   0 0 1:   1
E:1 0 0 0 0   0 0 1 = 129
```

Falls ein Bit in R oder in V (oder in beiden) gesetzt ist, wird dieses Bit auch als »1« in das Ergebnis übernommen. Umgekehrt werden jene Bits zu Null, deren entsprechenden Positionen in beiden Variablen zurückgesetzt sind.

Durch den OR-Befehl können gezielt Bits gesetzt werden. Im Bildschirmspeicher bewirkt das gesetzte siebte Bit, daß das Zeichen invertiert dargestellt wird. Schreiben Sie einmal ein Zeichen links oben auf den Bildschirm. Ermitteln Sie nun den »Z=PEEK(7680)« (PEEK(4096) beim erweiterten VC) den entsprechenden Zeichencode. Durch Z=Z OR 128 wird nun das siebente Bit gesetzt. POKE 7680,Z schreibt den Wert zurück und das Zeichen erscheint invertiert.

Das folgende kurze Programm invertiert den gesamten Bildschirm.

```
10 A=7680 : REM ODER 4096!
20 FOR I=0 TO 511 : D=PEEK(I+A)
30 D=D OR 128 : POKE A+I,D :
NEXT
```

# VIDEO CHIP 20

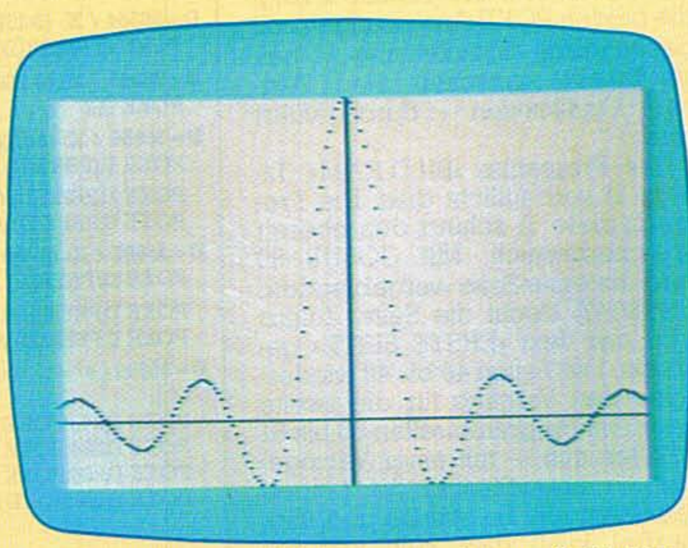


Bild 4. Ein weiteres Beispiel für die Anwendung der neuen Grafikbefehle

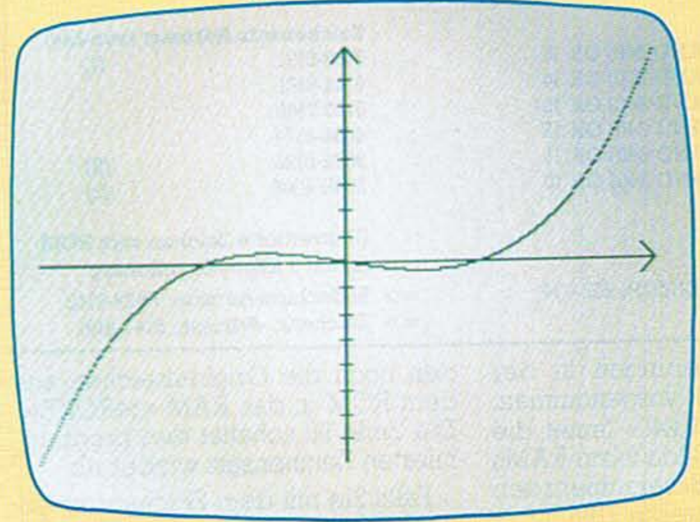


Bild 3. Das Plotten von Funktionen ist kein Problem mit der Grafikerweiterung

Falls Sie die Zeile 30 durch »D=D AND 127:...« ersetzen, werden die zuvor invertierten Zeichen wieder »normal« dargestellt. Durch die BIT-Schreibweise wird dies anschaulich:

```
R:1 0 0 0 0 0 0   1 : 129)
V:0 1 1 1 1 1 1   1 : 127)
E:0 0 0 0 0 0 0   1 =   1
```

Alle BIT-Positionen von R werden in das Ergebnis übernommen, sobald die entsprechenden Positionen des Vergleichswertes auf »1« gesetzt sind. Durch D AND 127 wird somit das siebte Bit in D gelöscht.

Die Tabelle 1 zeigt die für unsere Zwecke wichtigsten Register des Video-Chips sowie deren Funktion.

Die folgende Tabelle 2 verdeutlicht, wie die Register verändert werden können. Mit dem PEEK-Befehl wird der Inhalt der Speicheradresse abgefragt. Der AND-Befehl löscht nun einen Teilbereich und mit OR X wird zuletzt ein neuer Wert angeknüpft.

Außerdem gibt die Tabelle 2 die Originalwerte für die Variable X an, sowie — in Klammern — den zulässigen Zahlenbereich.

Die Adressen 36864/36865 zentrieren die Bilddarstellung auf dem Fernsehgerät oder Monitor. Mit dem Basicprogramm »BILD ZENTRIEREN« (Listing 1) können Sie dies testen. Der Bildausschnitt wird mit den Cursor-Tasten verschoben. Falls Sie die RETURN-Taste drücken, zeigt das Programm die notwendigen POKE-Befehle für die letzte Einstellung und setzt den Bildschirm zuletzt wieder in den Ausgangszustand zurück.

Das Programm »BILDSCHIRMFORMAT« (Listing 2) verändert die Adressen 36866/36867. Mit den Cursor-Tasten wird die Anzahl der Zeilen beziehungsweise die Anzahl der Zeichen pro Zeile verändert. Auch hier können Sie durch Drücken der RETURN-Taste die erforderlichen POKE-Befehle für die letzte Einstellung ermitteln.

Die Bilder 1 und 2 zeigen zwei geänderte Bildformate mit den dazugehörigen POKE-Befehlen.

## Zwischen verschiedenen Bildschirmbereichen umschalten

Das Register 36869 legt die Adressen für den Zeichensatz und den Bildschirmspeicher fest. Die Tabelle 2 zeigt vier POKE-Befehle mit den entsprechenden Bildschirm-Startadressen. Falls Sie eine Änderung durchführen, müssen Sie dem Computer die neue Startadresse in dem Zeiger 648 mitteilen. Außerdem müssen noch 25 weitere Zeiger (MSB der Zeilenanfänge) geändert werden. Durch SYS 58775 führt der Computer die notwendigen Berechnungen aus. Die POKE-Befehle in die Adressen 52/53 begrenzen den oberen Speicherbereich und schützen so das Bild-RAM vor dem Überschreiben durch Basic-Variablen.

Die mit einem Doppelkreuz »#« gekennzeichneten Adressen gelten, falls die Bildadresse zusätzlich um 512 Byte nach unten verschoben wurde. Diese »Feinverschiebung«

# Neues vom VC 20 Video Chip

erfolgt durch das siebte Bit der Adresse 36866. Die Tabelle 2 zeigt die beiden POKE-Anweisungen, mit denen diese Umschaltung — beispielsweise zwischen zwei Text- oder Grafikseiten — durchgeführt wird.

Das Programm »BILD-RAM« (Listing 3) verdeutlicht dies. Die Programmzeile 11 schützt den oberen Speicherbereich. Mit GOSUB 80 wird die erste Seite »vorgeblendet«. CHR$(147) löscht die Seite. Zuletzt wird der Text »ERSTE SEITE« gedruckt. Die Zeilen 40 bis 42 wiederholen den Vorgang für die zweite Seite. Die Programmzeilen 50 bis 61 schalten nun — mit einer Verzögerung — abwechselnd die beiden Testseiten ein. So schnell und flimmerfrei kann dies kein anderes Commodore-System.

Falls Sie nach dem Umschalten mit dem Register 36866 die Zeile »POKE 648,...:SYS 58775« weglassen, wird sich die Bildanzeige zwar ändern; PRINT-Befehle gelangen jedoch weiterhin in die alte — nun unsichtbare — Seite.

**Tabelle 2. So können die Register aus Tabelle 1 verändert werden**

| | |
|---|---|
| **D=36864 / X: (0-127) / Orig. 12** | : Horizontal-Position |
| POKE D, (PEEK(D) AND 128) OR X | : verschieben. |
| **D=36865 / X: (0-255) / Orig. 38:** | Vertikal-Position |
| POKE D,X | : verschieben. |
| **D=36866 / X: (0-27) / Orig. 150** | : Anzahl Zeichen |
| POKE D,(PEEK(D) AND 128) OR X | : pro Zeile. |
| POKE D,PEEK(D) OR 128 : oben | : Adresse des Bildschirm-RAM |
| POKE D,PEEK(D) AND 127 # unten | : um 512 Byte verschieben. |
| **D=36867 / X: (0-64) / Orig. 174** | |
| POKE D,(PEEK(D) AND 129) OR (2*X): | Anzahl der Zeilen ändern. |
| POKE D,PEEK(D) OR 1 (ein) | : Einfache oder doppelte |
| POKE D,PEEK(D) AND 254 (aus) | : Matrix einstellen. |
| **D=36869 (# => untere 512 Byte** | |
| | **Bildschirm-Adresse:** |
| POKE D,240:POKE 52/56/648,30 (#28) | : 7680 (#7168) |
| POKE D,224:POKE 52/56/648,26 (#24) | : 6656 (#6144) |
| POKE D,208:POKE 52/56/648,22 (#20) | : 5632 (#5120) |
| POKE D,192:POKE 52(56/648,18 (#16) | : 4608 (#4096) |
| | : |
| | **Zeichensatz-Adresse: (von-bis)** |
| POKE D,(PEEK(D) AND 240) OR 15 | : 7168-8191 (1) |
| POKE D,(PEEK(D) AND 240) OR 14 | : 6144-8181 |
| POKE D,(PEEK(D) AND 240) OR 13 | : 5120-7168 |
| POKE D,(PEEK(D) AND 240) OR 12 | : 4096-6144 |
| POKE D,(PEEK(D) AND 240) OR 11 | : 3072-5120 (2) |
| POKE D,(PEEK(D) AND 240) OR 10 | : 2048-4096 (2) |
| | : |
| | : (1) invertierte Zeichen vom ROM |
| | : (2) mit 3 KByte-Erweiterung |
| Beispiel: POKE D,PEEK(D), 208+14 | => Bildschirm-Adresse: 5632-6143 |
| | => Zeichens. -Adresse: 6144-8191 |

## Eigene Zeichen definieren

Aus der Tabelle 1 kann man ablesen, daß nur die oberen vier Bits der Adresse 36869 das Bild-RAM verschieben. Die restlichen Bits bestimmen die Lage des Zeichensatzes im Speicher. Durch die in der Tabelle 2 angegebenen OR-Verknüpfungen kann der Zeichensatz ins RAM gelegt werden. Nur »OR 15« stellt eine Ausnahme dar, da hierdurch die unteren 128 Zeichen aus dem RAM, die oberen — invertierten — Zeichen weiterhin aus dem ROM geholt werden. Diese Zeichen erscheinen dann aber »nichtinvertiert«.

In dem Programm »PROG. ZEICHENSATZ« (Listing 4) werden die notwendigen Änderungen in der Programmzeile 30 vorgenommen. Nach »POKE 36869,240« lautet die Startadresse des Bildschirm-RAMs 7680. Der Wert »+15« verschiebt den Zeichensatz zur Adresse 7168.

Im einzelnen schreibt das Programm den gesamten Zeichensatz des VC 20 auf den Bildschirm. Dabei werden die Zeichen durch »POKE 38400+I,0« in der Farbe Schwarz gedruckt. Die Zeile 30 verändert dann die Zeiger, so daß die ersten 128 Zeichen den zufälligen Inhalt des Zeichensatz-RAMs wiedergeben. Der Zeichensatzspeicher wird dann in den folgenden Zeilen gesetzt und anschließend gelöscht. Zuletzt werden noch die Originalzeichen aus dem ROM in das RAM »gePOKEt«. Die Zeile 55 schaltet den programmierten Zeichensatz wieder ab.

Falls Sie mit dem Zeichensatz experimentieren wollen, brauchen Sie nur die letzte Zeile wegzulassen. Alle »ungeshifteten« Zeichen können dann verändert werden. Zuerst müssen Sie jedoch ermitteln, wohin Sie »POKEn« müssen. Schreiben Sie hierzu das Zeichen, das Sie ändern wollen, links oben auf den Bildschirm. Mit »PRINT PEEK(7680)« kann dann der dazugehörige Code ermittelt werden. Der Wert muß nun mit acht multipliziert werden, da zu jedem Zeichen acht Byte gehören. Zuletzt müssen Sie noch die Startadresse des Zeichensatzes (hier 7168) hinzuaddieren. Für den Buchstaben »A« lautet die richtige Adresse 7174. POKEn Sie einmal den Wert 129 an diese Adresse. In Bit-Form sieht das wie folgt aus: 129 gibt 10000001.

Genauso wird anschließend die erste Zeile in dem Buchstaben »A« aussehen; rechts und links außen je ein Pünktchen. Verändert Sie auch einmal die nächsten Adressen oder

**Tabelle 1. Einige wenig bekannte, aber recht nützliche Register des VIC-Chips.**

| Adresse / BIT: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| 36864 = | — | HZ | HZ | HZ | HZ | HZ | HZ | HZ | : Horizontal-Zentrierung |
| 36865 = | VZ | VZ | VZ | VZ | VZ | VZ | VZ | VZ | : Vertikal-Zentrierung |
| 36866 = | BA | ZE | ZE | ZE | ZE | ZE | ZE | ZE | : Bildschirm-Adresse<br>: ZEichen pro Zeile |
| 36867 = | — | ZL | ZL | ZL | ZL | ZL | ZL | MA | : ZeiLen-Anzahl<br>: Matrix: (8*8 / 8*16) |
| 36869 = | BA | BA | BA | BA | ZA | ZA | ZA | ZA | : Bildschirm-Adresse<br>: Zeichensatz-Adresse |

versuchen Sie den Umlaut »Ä« zu gestalten.

Nach der Theorie nun ein praktisches Grafikbeispiel. Das abschreckend lange Basic-Ladeprogramm (Listing 5) erweitert den Basic-Befehlssatz des VC 20 (ohne Speichererweiterung) um sieben weitere Anweisungen (COPY, GNEW, GON, GOFF, PLOT, TURN und CLEAR). Beim Eintippen sollten Sie jedoch die Zeilen 10 bis 32 durch das Programm »Prüfsumme« (Listing 6) ersetzen. Nach RUN können Sie mit Hilfe der Tabelle 3 überprüfen, ob Sie die Daten fehlerfrei eingegeben haben. Die Prüfsummen gelten jeweils für eine DATA-Zeile.

Nachdem Sie das Ladeprogramm (Zeile 10 bis 32) hinzugenommen haben, muß das Programm unbedingt abgespeichert werden. Nach RUN sucht das Programm die Zeile 32 (also nicht weglassen) und überträgt die Daten ab dieser Adresse in den Speicher. In den Programmzeilen 22 bis 24 wird das Maschinenprogramm zuletzt in den oberen RAM-Bereich übertragen und durch die folgenden POKE-Befehle geschützt. Der NEW-Befehl löscht zuletzt das — ohnehin zerstörte — Programm.

Mit SYS 6273 können die Befehle jetzt eingeschaltet werden. Bei 2173 BYTES FREE nähert man sich natürlich gefährlich dem »Sinclair-Syndrom«.

Nach dem Einschalten fällt zuerst ein neuer Cursor auf. In der invertierten Blinkphase des Cursors wird ein »graues« Viereck (COMMODORE-SHIFT » + «) gedruckt. Dies ist notwendig, da der Zeichensatz keine invertierten Zeichen mehr umfaßt und deshalb der Cursor nicht mehr blinken würde. Durch das Grafikprogramm werden alle 64 umgeshifteten Zeichen programmierbar. Der Zeichensatzspeicher beginnt — wie in dem kleinen Textprogramm »PROG. ZEICHENSATZ« (Listing 4) — an der Adresse 7168. Mit dem Befehl »COPY« wird der Originalzeichensatz aus dem ROM in diesen Bereich kopiert.

Der Befehl »GON« schaltet nun den Grafik-Modus ein. Da zuvor der Originalzeichensatz ins RAM kopiert wurde, werden sich die ersten 64 Zeichen nicht verändern. Falls invertierte Zeichen auf dem Bildschirm stehen, werden diese jetzt nichtinvertiert wiedergegeben. Nur bei den geshifteten Zeichen (außer den invertierten) gibt es einen Totalausfall, aber bei 3,5 KByte Speicherraum muß man schon gewisse Mängel in Kauf nehmen. Mit »GOFF« wird der Zeichensatz wieder ins ROM verlegt, so daß alle Zeichen sichtbar werden.

Die drei Befehle PLOT/CLEAR/TURN setzen, löschen oder invertieren einen Bildpunkt. Natürlich muß hinter den Befehlen eine X- und Y-Koordinate angegeben werden. Beispielsweise »PLOT 176,1«; allerdings führt diese Eingabe zu der Fehlermeldung X-ERROR (IN...), da der zulässige Wertebereich für die X-Koordinate 0 bis 175 (22 Zeichen * 8 — 1), und für die Y-Koordinate 0 bis 183 (23 Zeichen * 8 — 1) beträgt.

Der Wertebereich für die Koordinatenangabe wird durch den letzten Befehl »GNEW« ermittelt, da das Bildschirmformat — wie zuvor beschrieben — per Software geändert werden kann. Bei einer Einstellung mit beispielsweise 25 Zeichen pro Zeile kann der Wert für X zwischen 0 und 199 variieren.

Außerdem löscht der GNEW-Befehl die 64 programmierbaren Zeichen sowie alle nichtinvertierten Zeichen auf dem Bildschirm. Da sich eine Grafik jedoch aus genau solchen Zeichen zusammensetzt, wird — falls sich eine Zeichnung auf dem Bildschirm befindet — diese gelöscht. Invertierte Zeichen werden hingeben nicht verändert, so daß Texte oder Beschriftungen erhalten bleiben.

In den Zeilen 10 bis 20 wird der Bildschirm gelöscht und der invertierte Text »TEXT-PROGRAMME« geschrieben. Der Befehl »GNEW« löscht den Grafikspeicher und den Bildschirm, wobei der invertierte Text unverändert bleibt. GON schaltet dann die Grafik ein und in der Schleife wird eine Linie von links unten nach rechts oben (in der augenblicklichen Schriftfarbe) gezeichnet. Sobald Sie eine Taste drücken, wird die Grafik wieder abgeschaltet und Sie sehen eine schräge Linie aus Fragezeichen.

Das Maschinenprogramm schreibt ein programmierbares Zeichen auf den Bildschirm und ändert den Grafikinhalt dieses Zeichens. Anschließend wird geprüft, ob schon ein Zeichen mit genau dieser Grafik umdefiniert wurde. Ist dies der Fall, so wird das gefundene Zeichen auf den Bildschirm gebracht und das alte Zeichen ist wieder frei. Wegen dieser Optimierungs-Routine »verbraucht« das Beispielprogramm nur ein Zeichen aus dem schmalen Vorrat von 64 programmierbaren Zeichen.

Da diese Optimierung über einen Vektor verläuft, kann sie durch »POKE 1,PEEK(1)—1« abgeschaltet werden. Lassen Sie das Testprogramm erneut laufen und sehen Sie sich das neue Ergebnis an. Mit »POKE 1,PEEK(1)+1« optimiert das Programm wieder, wird aber auch etwas langsamer.

Durch »PEEK(0)« können Sie außerdem abfragen, wie viele Zeichen noch frei sind. Hierdurch kann die Fehlermeldung »CHARACTER-ERROR (IN...)« verhindert werden. Falls PEEK(0) den Wert Null ergibt, sollten keine Grafikbefehle mehr folgen.

**Tabelle 3. Prüfsummen zum Grafikprogramm**

| | | | | | | |
|---|---|---|---|---|---|---|
| Zeile 50-54 | : | 2390 | 2436 | 2823 | 2640 | 3449 |
| Zeile 55-59 | : | 3487 | 3010 | 3321 | 2882 | 2608 |
| Zeile 60-64 | : | 2930 | 2741 | 2458 | 3089 | 2285 |
| Zeile 65-69 | : | 2480 | 2156 | 2935 | 3048 | 2241 |
| Zeile 70-74 | : | 2495 | 2564 | 3256 | 2723 | 2393 |
| Zeile 75-79 | : | 1864 | 2230 | 2187 | 2148 | 3300 |
| Zeile 80-84 | : | 3135 | 2640 | 3023 | 2734 | 1918 |
| Zeile 85-88 | : | 2258 | 1431 | 1564 | 449 | — |

Doch nun ein Beispiel:

```
10 PRINT CHR$(147);CHR$(18);
20 PRINT »TEST-PROGRAMM«
30 GNEW
40 GON
50 FOR I=0 TO 111
60 : PLOT I,I
70 NEXT
80 POKE 198,0:WAIT 198,1
90 GOFF
```

Ändern Sie nun die Zeile 60 des Testprogramms in »60 TURN I,I« und erweitern Sie folgende Zeile mit »GOTO 50«.

# NEUES VOM VC 20 VIDEO CHIP

Das Programm wird jetzt eine Linie ziehen, anschließend löschen und so fort, da die Bildpunkte jetzt fortlaufend invertiert werden. Falls Sie in der Zeile 40 den Befehle »GON« durch »GOFF« ersetzen, können Sie anschaulich verfolgen, wie das Programm — und speziell die Optimierung — arbeitet.

Eine Besonderheit müssen Sie jedoch unbedingt beachten. Die neuen Basicbefehle können nicht direkt hinter einer THEN-Anweisung stehen. Am einfachsten ist es, die beiden Befehle dann durch einen Doppelpunkt (...THEN : COPY) zu trennen.

**Listing 1. »Bild zentrieren«**

```
10 REM ******** BILD ZENTRIEREN *********
15 :
20 H=12         : V=38
25 POKE 198,0 : WAIT 198,1 : GET A$
30 IF A$=CHR$( 13)             THEN 60
35 IF A$=CHR$( 17) AND V<255 THEN V=V+1
40 IF A$=CHR$(145) AND V>0   THEN V=V-1
45 IF A$=CHR$( 29) AND H<23  THEN H=H+1
50 IF A$=CHR$(157) AND H>1   THEN H=H-1
55 POKE 36864,H :POKE 36865,V:GOTO 25
60 POKE 36864,12:POKE 36865,38
65 PRINT "POKE 36864 ,";H
70 PRINT "POKE 36865 ,";V
READY.
```

Zuletzt eine Grafik-Anwendung. Die Bilder 3 und 4 zeigen, welche Ergebnisse mit der Grafik-Erweiterung möglich sind. Das Basicprogramm »FUNKTIONS-PLOT« (Listing 7) vereinfacht das programmierte Plotten von Funktionen doch erheblich.

Nachdem Sie das Programm eingegeben und gestartet haben, werden Sie nach dem Wertebereich gefragt. Anschließend wird die Bildumrahmung gezeichnet. Die Größe sowie die Position des Diagramms auf dem Bildschirm ist durch die Variablen Unten, Oben, Rechts und Links (U,O,R,L) in der Zeile 110 festgelegt. Die Zeilen 170 bis 230 berechnen nun das Minimum sowie das Maximum der Funktion. Außerdem werden die Werte in das Feld Y(A) übertragen. Die Variablen MI und MA werden in Zeile 250 auf vier Nachkommastellen begrenzt. Diese Werte sowie die Funktion selbst werden nun (invertiert) im unteren Bildteil gedruckt. Nachdem die Funktion grafisch dargestellt wurde, läuft das Programm in Zeile 390 in einer Warteschleife.

Drücken Sie jetzt eine Taste, so gelangen Sie in ein Mini-Menü. Sie können nun eine neue Funktion eingeben oder die alte mit neuen Werten untersuchen.

Eine neue Funktion wird in der Zeile 450 mit INPUT A$ übergeben. Salopp gesagt, programmiert sich der Copmputer in den folgenden Zeilen selbst. Wie das?

Zuerst wird der Bildschirm gelöscht und in der obersten Zeile die Zeilennummer 190 und der String A$, der die neue Funktion enthält, gedruckt. Die Zeile 460 druckt darunter die Nummer 190 und einen PRINT-Befehl. Anschließend folgt ein Anführungszeichen, dann die Funktion und schließlich wieder ein Anführungszeichen. Zuletzt wird in der dritten Bildschirmzeile der Befehl »RUN« gedruckt. Jetzt könnte auf dem Bildschirm der folgende Text stehen:

```
190 Y=COS(X)
290 ?"Y=COS(X)"
RUN
```

Wäre das Programm hier zu Ende, so müßten Sie nur die HOME-Taste drücken und der Cursor würde auf der Zeile 190 stehen. Drückt man jetzt RETURN, so wird diese Zeile

**Listing 2. »Bildschirmformat«**

```
10 REM ****** BILDSCHIRM-FORMAT *******
15 :
20 V=46         : H=22+128
25 POKE 198,0 : WAIT 198,1 : GET A$
30 IF A$=CHR$( 13)             THEN 60
35 IF A$=CHR$( 17) AND V<100 THEN V=V+2
40 IF A$=CHR$(145) AND V>2   THEN V=V-2
45 IF A$=CHR$( 29) AND H<155 THEN H=H+1
50 IF A$=CHR$(157) AND H>128 THEN H=H-1
55 POKE 36866,H :POKE 36867,V  :GOTO 25
60 POKE 36867,46:POKE 36866,150
65 PRINT "POKE 36866 ,";H
70 PRINT "POKE 36867 ,";V
READY.
```

**Listing 3. »Bild-RAM« — Umschalten zwischen zwei Bildschirmseiten**

```
1 REM ***** BILD-RAM *****
2 :
10 D=36866:P1=30:P2=28
11 POKE 52,P2:POKE 56,P2
15 :
30 GOSUB 80
31 PRINT CHR$(147):PRINT
32 PRINT" ERSTE SEITE"
35 :
40 GOSUB 90
41 PRINT CHR$(147):PRINT
42 PRINT" ZWEITE SEITE"
45 :
50 FOR Z=0 TO 300:NEXT
51 GOSUB 80
60 FOR Z=0 TO 300:NEXT
61 GOSUB 90       :GOTO 50
70 :
80 POKE D,PEEK(D) OR 128
81 POKE 648,P1:SYS 58775
82 RETURN
85 :
90 POKE D,PEEK(D) AND 127
91 POKE 648,P2:SYS 58775
93 RETURN
READY.
```

einprogrammiert. Ein erneuertes RETURN übernimmt die Zeile 290 in das Programm und beim dritten RETURN wird der Befehl RUN ausgeführt. Da sich der Computer bis zu neun Tastatureingaben »merken« kann, brauchen Sie diesen Ablauf nicht »von Hand« einzugeben. Der erste POKE-Befehl in der Zeile 470 sagt dem Computer, daß — angeblich — sechs Tasten gedrückt worden sind. Zuerst »CURSOR HOME« (POKE 631,19), und dann fünfmal »RETURN« (Zeile 480). Nach dem END-Befehl führt er diese — untergeschobenen — Eingaben aus, programmiert die beiden Programmzeilen ein und startet sich zuletzt wieder mit dem RUN-Befehl.

Dieses Verfahren ist übrigens sehr praktisch, falls man ein Maschinenprogramm in DATA-Zeilen umwandeln muß. Vorausgesetzt das notwendige Programm ist fehlerfrei, kann man eigentlich sicher sein, daß es der Basic-Lader anschließend auch ist.

Wie Sie an dem Funktions-Plotter sehen können, ermöglicht die Grafikhilfe die Programmierung auch komplexerer Grafiken in einfacher und überschaubarer Form. Da die Erstellung der Grafik recht schnell erfolgt, ist eine große Vielfalt von Anwendungen denkbar.

(Heino Verder/ev)

**Listing 4. »Programmierbarer Zeichensatz«**

```
10 REM ******* PROG. ZEICHENSATZ ******
11 :
12 POKE 52,26             :POKE 56,26
15 :
20 FOR I=0 TO 255        :POKE 7680+I,I
25 POKE 38400+I,0        :NEXT
30 POKE 36869,240+15
35 FOR I=7168 TO 7679 :POKE I,255:NEXT
40 FOR I=7168 TO 7679 :POKE I,0   :NEXT
45 FOR I=0    TO  511 :C=PEEK(32768+I)
50 POKE 7168+I,C         :NEXT
55 POKE 36869,240+0
READY.
```

**Listing 6. Prüfsummenprogramm zum Basic-Lader aus Listing 5. Das Zeichen » â « in Zeile 28 ist der Hochpfeil (»↑«)**

```
10 REM ********** PRUEFSUMME **********
12 ZD=50:A=I:FOR I=0 TO 896:READ A$
14 ZL=PEEK(60)+PEEK(61)*256:D=0
16 IF ZL=ZD THEN 20
18 PRINT "ZEILE";ZD;C : ZD=ZL : C=0
20 FOR L=1 TO 2:IF LEN(A$)<2THEN A$="0"+A$
22 W$=MID$(A$,L,1):W=0:IF W$=""THEN 24
24 W=ASC(W$)-48   :IF W>9 THEN W=W-7
26 :
28 D=D+W*16^(2-L) :NEXT : C=C+D : S=S+1
30 NEXT : PRINT "ZEILE";ZD,C    : END
32 :
50 DATA EA,EA,EA,78,A9,BC,BD,14,3,A9...
READY.
```

# NEUES VOM VC 20 VIDEO CHIP

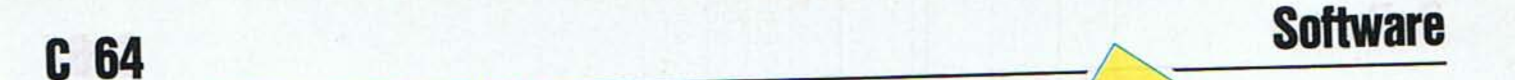
**Listing 5. Basic-Ladeprogramm zur Befehlserweiterung**

```
10 FOR I=4096 TO 9999:IF PEEK(I)<>42THEN NEXT
12 A=I:FOR I=0 TO 896:READ A$:D=0:PRINT"":I
14 FOR L=1 TO 2:IF LEN(A$)<2THEN A$="0"+A$
16 W$=MID$(A$,L,1):W=0:IF W$=""THEN 18
18 W=ASC(W$)-48   :IF W>9 THEN W=W-7
20 :
22 D=D+W*16^(2-L) :NEXT:POKE I+A,D:NEXT
24 FOR I=0 TO 896:D=PEEK(A+I):POKE 6272+I,D:NEXT
26 POKE 51,128 : POKE 52,24
28 POKE 55,128 : POKE 56,24  :NEW
30 :
32 REM **********
50 DATA EA,EA,EA,78,A9,BC,8D,14,3,A9,18,8D,15,3,A9,18,8D,8,3,A9,19,8D,9
51 DATA 3,A9,11,85,1,A9,1B,85,2,8D,F2,1B,58,60,78,A9,BF,8D,14,3,A9,EA,8D
52 DATA 15,3,A9,E4,8D,8,3,A9,C7,8D,9,3,58,60,AD,F2,1B,F0,D,CE,F2,1B,A4,D3
53 DATA B1,D1,8D,F1,1B,8D,F0,1B,A9,18,48,A9,DB,48,8,8,8,8,4C,BF,EA,AE,5
54 DATA 90,E8,D0,0,A4,D3,A6,CC,D0,2E,AD,F1,1B,F0,8,8E,F1,1B,B1,D1,8D,F0
55 DATA 1B,A5,C6,F0,E,A5,CB,C9,1B,F0,5,EE,F2,1B,D0,9,EE,F1,1B,A9,E6,A6,CF
56 DATA D0,3,AD,F0,1B,D1,D1,F0,2,91,D1,4C,18,EB,A9,7,85,58,A9,0,48,AA,CA
57 DATA A0,0,C8,E8,BD,AB,1B,F0,F,D1,7A,F0,F5,18,68,69,8,C6,58,D0,E8,4C,E4
58 DATA C7,68,88,98,18,65,7A,85,7A,90,2,E6,7B,BD,AC,1B,85,F7,BD,AD,1B,85
59 DATA F8,20,55,19,4C,18,19,6C,F7,0,A2,FF,D0,2,A2,F0,8E,5,90,60,A0,0,A2
60 DATA 2,84,62,84,64,A9,80,85,63,A9,1C,85,65,B1,62,91,64,C8,D0,F9,E6,63
61 DATA E6,65,CA,D0,F2,60,AD,2,90,29,7F,8D,F5,1B,A,A,A,8D,F4,1B,AD,3,90
62 DATA 29,7E,A,A,8D,F3,1B,CE,F3,1B,A9,1C,85,63,A9,1E,85,65,A0,0,84,62,84
63 DATA 64,A2,1,A9,0,91,62,B1,64,30,4,A9,A0,91,64,C8,D0,F1,E6,63,E6,65,CA
64 DATA 10,EA,A2,3F,86,0,A2,7F,A9,0,9D,50,3,CA,10,FA,60,A2,0,2C,A2,2,2C
65 DATA A2,4,BD,E0,1B,8D,40,3,BD,E1,1B,8D,41,3,20,73,0,20,8A,CD,20,F7,D7
66 DATA A0,58,A6,15,D0,29,8E,43,3,A5,14,8D,42,3,CD,F4,1B,B0,1C,20,79,0,20
67 DATA FD,CE,20,8A,CD,20,F7,D7,A5,15,D0,8,AD,F3,1B,38,E5,14,B0,17,A9,59
68 DATA D0,2,A9,58,20,D2,FF,A9,2D,20,D2,FF,20,7A,C6,C8,A9,6B,4C,67,C4,AA
69 DATA 20,98,1B,8E,45,3,A9,1E,85,5B,A9,0,C0,0,F0,B,18,6D,F5,1B,90,2,E6
70 DATA 5B,88,D0,F5,85,5A,AE,42,3,20,98,1B,AD,43,3,F0,5,98,18,69,20,A8,A9
71 DATA 0,E8,38,6A,CA,D0,FC,86,FD,8D,44,3,B1,5A,8D,47,3,30,15,C5,0,90,11
72 DATA AA,BD,50,3,D0,6,B1,5A,C5,0,B0,3A,DE,50,3,E6,FD,A5,FB,F0,9,C6,FB
73 DATA A6,FB,BD,E8,3,D0,11,A5,0,D0,B,58,A9,E6,A0,1B,20,1E,CB,4C,22,1A,C6
74 DATA 0,91,5A,AA,A5,5A,18,69,0,85,58,A5,5B,69,78,85,59,AD,86,2,91,58,8A
75 DATA 78,85,FC,8C,46,3,A2,0,86,59,A,26,59,A,26,59,A,26,59,18,69,0,85,58
76 DATA A5,59,69,1C,85,59,A5,FD,F0,24,A9,0,85,5E,AD,47,3,A,26,5E,A,26,5E
77 DATA A,26,5E,18,69,0,85,5D,A5,5E,69,1C,85,5E,A0,7,B1,5D,91,58,88,10,F9
78 DATA AC,45,3,AD,44,3,20,8A,1B,91,58,58,6C,1,0,60,78,A0,7,84,FD,B1,58
79 DATA 99,E0,3,D0,2,C6,FD,88,10,F4,E6,FD,D0,4,A9,A0,D0,24,A9,1E,85,5E,A9
80 DATA 0,85,5D,A2,40,D0,42,A0,7,E4,FC,F0,3C,E4,0,F0,48,B1,5D,D9,E0,3,D0
81 DATA 31,88,10,F6,FE,50,3,8A,AC,46,3,91,5A,A6,FC,BD,50,3,F0,3,DE,50,3
82 DATA 8A,E6,0,C5,0,F0,9,C6,0,A6,FB,9D,E8,3,E6,FB,A0,7,A9,0,91,58,88,10
83 DATA FB,30,10,A5,5D,38,E9,8,B0,2,C6,5E,85,5D,CA,E4,0,D0,AE,58,60,6C,40
84 DATA 3,11,58,60,51,58,60,49,FF,31,58,60,A0,0,8A,10,4,A0,10,29,7F,AA,38
85 DATA E9,8,90,3,C8,D0,F7,60,50,4C,4F,54,0,D1,19,0,54,55,52,4E,0,D4,19
86 DATA 0,43,4C,45,41,52,0,D7,19,43,4F,50,59,0,62,19,0,CB,4E,0,58,19,0,0
87 DATA 0,CB,46,46,0,5C,19,0,0,47,A2,0,81,19,8D,1B,90,1B,93,1B,43,48,41
88 DATA 52,41,43,54,45,52,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
READY.
```

**Listing 7. Basicprogramm »Funktions-Plot« zur Demonstration der neuen Grafikbefehle**

```
CURSOR-CODE:  PRINT"C"  =>  CLEAR HOME
                   "H"  =>  CURS. HOME
                   "D"  =>  CURS. DOWN
                   "R"  =>  REVERSE

100 CLR:INPUT"CR WERTEBEREICH";X1,X2
110 U=80:O=183:L=0:R=175:D=R-L:K=10000
120 GNEW:GON:DIM Y(D):IF X1=X2 THEN 100
130 FOR A=L TO R:PLOT A,U:PLOT A,O:NEXT
140 FOR A=U TO O:PLOT L,A:PLOT R,A:NEXT
150 IF X1>X2 THEN Z=X1:X1=X2:X2=Z
160 :
170 SX=(X2-X1)/D:X=X1:MI=1E38:P=MI:MA=-MI
180 FOR A=0 TO D
190 Y=SIN(X)+SIN(2*X)
200 IF Y>MA THEN MA=Y
210 IF Y<MI THEN MI=Y
220 IF ABS(Y)<P THEN P=ABS(Y):N=A
230 Y(A)=Y:X=X+SX:NEXT:SY=(MA-MI)/(O-U-5)
240 :
250 MI=INT(MI*K)/K:MA=INT(MA*K)/K:K=0
260 PRINT "SDDDDDDDDDDDDDDDR  ";:T1=K
270 T2=LEN(STR$(X1))-LEN(STR$(MI))
280 IF T2<K THEN T1=-T2:T2=K
290 PRINT "Y=SIN(X)+SIN(2*X)"
300 PRINT TAB(T1);"RD";X1;"<X<";X2
310 PRINT TAB(T2);"R" ;MI;"<Y<";MA
320 IF MI=>K AND MA=>K OR MI<=K AND MA<=K THEN 340
330 P=U+3+INT((Y(N)-MI)/SY):FOR A=L TO R:PLOT A,P:NEXT
340 IF X1=>K AND X2=>K OR X1<=K AND X2 <=K THEN 360
350 P=INT(ABS(X1)/SX):FOR A=U TO O:PLOT P,A:NEXT
360 :
370 FOR A=L TO R:P=U+3+INT((Y(A-L)-MI)/SY)
380 PLOT A,P:IF PEEK(K) THEN NEXT
390 POKE 198,0:WAIT 198,1:GET A$
400 PRINT "CR F => FUNKTION"
410 PRINT "R W => WERTEBEREICH"
420 WAIT 198,1:GET A$
430 IF A$= "W" THEN 100
440 IF A$<>"F" THEN 420
450 COPY:INPUT "C FUNKTION";A$:PRINT "C 190";A$
460 PRINT"290 ?";CHR$(34);A$;CHR$(34)
470 PRINT "RUN ":POKE 198,6:POKE 631,19
480 FOR I=632 TO 636:POKE I,13:NEXT:END
READY.
```







# Datenkreislauf

# Die sequentielle Datenspeicherung

**Als es noch keine Diskettenlaufwerke gab, standen lediglich Bandgeräte zur Datenspeicherung zur Verfügung. Und auf Bändern ist nur sequentielles Speichern möglich. Auch mit der Floppy kann man Daten sequentiell speichern, obwohl der Vorteil der Diskette, der direkte Zugriff auf Daten, nicht ausgenutzt wird.**

Wenn Sie sich das Diskettenlaufwerk gerade erst gekauft haben, werden Sie sicher zu Beginn einige Schwierigkeiten haben, es auch richtig einzusetzen. Das erste, was man macht, wird das Laden und Abspeichern von Programmen sein. Daß Sie mit der Floppy noch viel mehr machen können, wissen Sie ja auch schon längst. Und sicher fallen Ihnen einige Möglichkeiten ein, die Sie wohl gerne realisieren möchten. Meistens sind das am Anfang eine Adressenkartei, eine Schallplattensammlung erfassen oder etwa seine Dias ordnen und auswerten. Wir möchten Ihnen den Einstieg etwas erleichtern und Sie Schritt für Schritt mit den Methoden der Datenverwaltung vertraut machen.

Es gibt verschiedene Arten von Dateien und unterschiedliche Methoden ihrer Verwaltung. Hauptsächlich sind das die sequentielle, die relative Datei und der Direktzugriff auf die Diskette.

Gerade für Anfänger ist die sequentielle Datei eine geeignete Methode, Daten zu verwalten. Diese Einfachheit ist allerdings verbunden mit einer gewissen Langsamkeit. Aber wenn sie sich erst einmal mit der sequentiellen Datenspeicherung auskennen, wird auch die relative Datei und die Direktzugriffsmethode kein großes Problem mehr für Sie darstellen. Außerdem gibt es genügend Probleme, für die eine sequentielle Datenspeicherung völlig ausreichend ist.

Zuerst sollen Sie einige wichtige Begriffe kennenlernen.

Sequentiell bedeutet hintereinander. Also alle Daten, die wir benutzen, werden hintereinander, sequentiell auf der Diskette gespeichert (bei Kassetten ist eine andere Methode gar nicht möglich).

Immer, wenn man von und über Dateien spricht, werden solche Vokabeln wie FELD, DATENSATZ oder auch RECORDS genannt. Was ist das eigentlich?

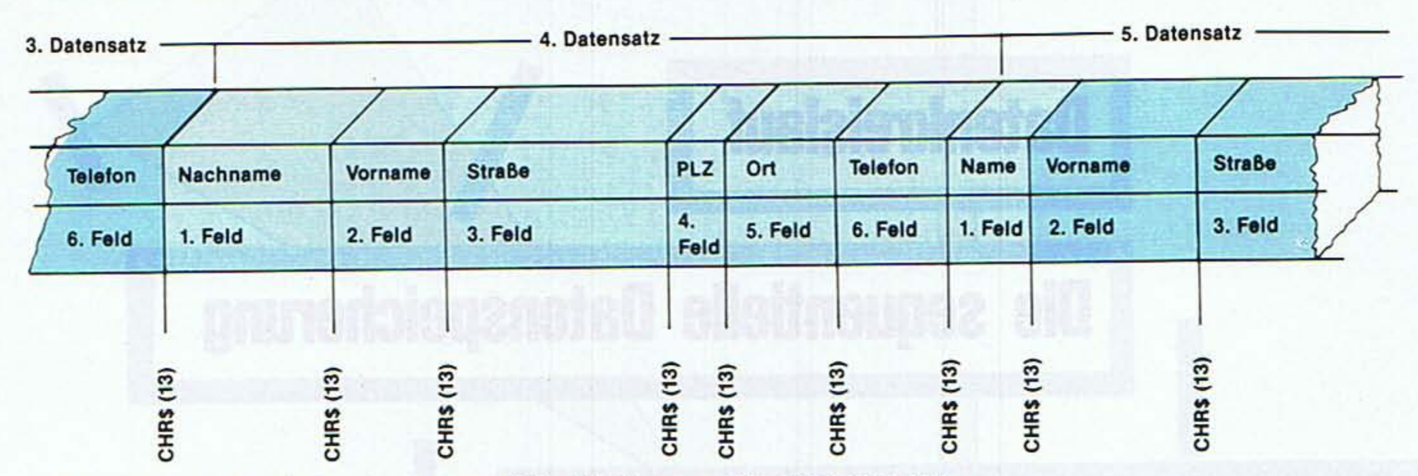
## Der Computer ersetzt Karteikarten

Die meisten von Ihnen kennen eine Karteikarte. Auf dieser Karteikarte können mehrere Eintragungen stehen. Angenommen, Sie besitzen eine Adressenkartei, dann stehen auf jeder Karteikarte normalerweise der Vorname, Nachname, Straße, Postleitzahl und Ort und die Telefonnummer, manchmal auch der Geburtstag und andere wichtige Daten. Das sind die einzelnen Felder eines Datensatzes. Die Gesamtheit aller Felder auf einer Karteikarte bezeichnet man als einen Datensatz, also alle Daten, die zusammengehören. Somit besteht ein Datensatz aus mehreren Feldern. Und der Begriff Record ist nur die englische Bezeichnung für Datensatz. Alle Karteikarten, also alle Datensätze, nennt man eine Datei. Damit wären die wichtigsten Begriffe geklärt und wir können zum Wesen der sequentiellen Datei kommen.

Wie schon gesagt, werden bei dieser Dateiart alle Daten hintereinander (sequentiell) auf der Diskette gespeichert. Das sieht dann zum Beispiel so aus wie in Bild 1.

Angenommen, unsere Datei besitzt schon 200 Adressen und eine Adresse enthält den Namen Huber, Anton. Wollen wir jetzt seine Anschrift wissen, so muß die Datei von Anfang an durchsucht werden, bis zu dem Datensatz, in der der Huber steht. Steht der gesuchte Name ziemlich am Ende der Datei, so kann die Suche einige Zeit in Anspruch nehmen; je größer die Datei, desto mehr Zeit beansprucht die Suche. Diese Suche, aber auch alle anderen Operationen wie Ergänzungen, Veränderungen oder Sortieren der Datei spielen sich im Speicher des Computers ab. Das bedeutet, daß, wenn man mit dieser Adressendatei arbeiten will, zuerst die gesamte Datei von der Diskette in den Speicher geladen werden muß. Und am Ende, nach getaner Arbeit, wird sie wieder auf die Diskette zurückgespeichert.

Wenden wir uns dem Beispiel in Bild 2 zu. Es ist ein kleines und noch nicht einmal fertiges Adressen-Datei-Programm (die Funktionen »Suchen«, »Löschen« und »Ändern«) sind noch nicht codiert worden. Jedoch kann sich jeder dieser Module selber schreiben. Auf eine Besonderheit möchte ich gleich hinweisen. In den Zeilen 133 bis 135 stehen die Erklärungen der verwendeten Steuerzeichen, die diesmal codiert ausgedruckt wurden. Deswegen




**In dieser Reihenfolge stehen die Daten in einer sequentiellen Datei (bezogen auf das Beispielprogramm)**

brauchen Sie nicht mehr mit der Lupe und viel Fantasie, die manchmal schwierig zu erkennenden Commodore-Steuerzeichen versuchen, zu interpretieren. Drücken Sie einfach die entsprechende Cursortaste (innerhalb der PRINT-Anweisung) wenn die Zeichen »CRR« und »CRL« im Listing erscheinen. Auf Ihrem Bildschirm sollten sie dann die entsprechenden Grafikzeichen erkennen. Gefällt Ihnen diese Art der Darstellung? Wir würden gerne Ihre Meinung dazu erfahren.

Der Aufbau des Programms dürfte nicht schwer zu durchblicken sein. Aber es gibt einige Programmzeilen und -abschnitte, die man sich einmal näher anschauen sollte.

Zeile 9110: Die Variable ST ist eine Statusvariable, die der C 64 und der VC 20 für sich beanspruchen. Hier gibt sie Auskunft darüber, ob das Dateiende erreicht ist. In diesem Fall wird nämlich das 6. Bit gesetzt (2 hoch 6 ergibt 64). Da aber in dieser Systemvariablen auch andere Nachrichten abgelegt werden, die andere Bits beeinflussen und ST auch bei Dateiende einen von 64 ungleichen Wert zuweisen können, müssen wir mit der AND-Verknüpfung feststellen, ob speziell dieses 6. Bit gesetzt ist.

Aus Sicherheitsgründen habe ich vor jedem OPEN-Befehl ein CLOSE

```
100 REM  ****************************
110 REM  * ADRESSENDATEI 64'ER/6     *
120 REM  ****************************
130 :
131 REM  ***** STEUERZEICHEN *****
132 REM  *                        *
133 REM  * <CLR> = CLR/HOME       *
134 REM  * <CRR> = CURSOR RECHTS  *
135 REM  * <CRL> = CURSOR LINKS   *
136 REM  *                        *
137 REM  **************************
138 :
140 DI=200
150 DIM NV$(DI),NN$(DI),SR$(DI)
160 DIM PL$(DI),OT$(DI),TE$(DI)
170 POKE 53281,0
180 CLOSE 15:OPEN 15,8,15
1000 REM  ---------------------------
1010 REM  - MENUE                   -
1020 REM  ---------------------------
1030 :
1040 PRINT "<CLR>"
1050 PRINT :PRINT
1060 PRINT "      ADRESSENDATEI"
1070 PRINT :PRINT
1080 PRINT "    X = PROGRAMMENDE"
1090 PRINT
1100 PRINT "    1 = ANZEIGEN GESAMTE DATE
I"
1110 PRINT
1120 PRINT "    2 = SUCHEN"
1130 PRINT
1140 PRINT "    3 = LOESCHEN"
1150 PRINT
1160 PRINT "    4 = AENDERN"
1170 PRINT
1180 PRINT "    5 = NEUE ADRESSEN EINGEBE
N"
1190 PRINT
1200 PRINT "    6 = SPEICHERN"
1210 PRINT :PRINT :PRINT :PRINT
1220 PRINT "WAEHLEN SIE ";
1230 GET R$:IF R$="" THEN 1230
1240 IF R$="X" THEN 1500
1250 IF R$="1" THEN GOSUB 2000:REM  ANZ.
1260 IF R$="2" THEN GOSUB 3000:REM  SUCH
1270 IF R$="3" THEN GOSUB 4000:REM  DEL
1280 IF R$="4" THEN GOSUB 5000:REM  AEN
1290 IF R$="5" THEN GOSUB 6000:REM  NEU
1300 IF R$="6" THEN GOSUB 8000:REM  SPEI
1310 GOTO 1000
1320 :
1500 REM  ---------------------------
1510 REM  - PROGRAMM ENDE           -
1520 REM  ---------------------------
1530 REM  SS=2 WENN NICHT GESPEICHERT
1540 IF SS=1 THEN GOSUB 8000
1550 CLOSE 15
1560 END
1570 :
1580 :
2000 REM  ------------------------------
2010 REM  - ANZEIGEN GESAMTE DATEI
2020 REM  ------------------------------
2030 :
2040 IF L=0 THEN GOSUB 9000:REM  LADEN
2050 FOR I=1 TO ND
2060 :PRINT "<CLR>    ANZEIGE DER DATEI
"
2065 :PRINT
2070 :PRINT " NUMMER ";I:PRINT
2075 :PRINT
2080 :PRINT " NAME     : "NV$(I)" "NN$(I)
2100 :PRINT " STRASSE  : "SR$(I)
2110 :PRINT " PLZ/ORT  : "PL$(I)" "OT$(I)
2130 :PRINT " TELEFON  : "TE$(I)
2150 :PRINT :PRINT :PRINT
2160 :PRINT "  W = WEITER      X = ENDE"
2170 :GET R$:IF R$="" THEN 2170
2180 :IF R$="X" THEN 2200
2190 NEXT I
2200 RETURN
2210 :
6000 REM ------------------------------
6010 REM - EINGABE NEUE DATEN          -
6020 REM ------------------------------
6030 :
6040 IF L=0 THEN GOSUB 9000:REM  LADEN
6050 PRINT "<CLR>"
6060 PRINT "     EINGABE NEUE DATEN"
6070 PRINT :PRINT
6075 ND=ND+1
6076 PRINT "     NUMMER ";ND
6078 PRINT :PRINT
6080 INPUT "NACHNAME <CRR><CRR><CRR>.<CR
L><CRL><CRL>";NN$(ND)
6090 INPUT "VORNAME  <CRR><CRR><CRR>.<CR
L><CRL><CRL>";NV$(ND)
6100 INPUT "STRASSE  <CRR><CRR><CRR>.<CR
L><CRL><CRL>";SR$(ND)
6110 INPUT "POSTLZ.  <CRR><CRR><CRR>.<CR
L><CRL><CRL>";PL$(ND)
```

**Bild 2. Kleine Adressdatei**

gesetzt, damit nicht durch einen Programmierfehler ein »FILE OPEN ERROR« entstehen kann. Zum CLOSE-Befehl sei noch so viel gesagt, daß er unbedingt existieren muß. Sonst kann es vorkommen, daß nicht alle Daten auf die Diskette übertragen werden, und noch schlimmer, daß die Datei nicht ordnungsgemäß geschlossen werden kann. Dadurch kann die gesamte Datei unrettbar verloren sein. Also Vorsicht.

Wenn Sie sich das Programm etwas näher anschauen, könnten Sie über die Variablen »SS« und »L« stolpern. Diese Variablen habe ich eingesetzt als »Flags«. Sie regeln, ob ein Aufruf dieses Menüpunktes sinnvoll ist und das entsprechende Unterprogramm ausgeführt werden soll. Zum Beispiel verhindert die Variable »L« in Zeile 8025, daß Daten gespeichert werden, wenn die Datei noch gar nicht in den Speicher des Computers geladen wurde. Das würde nämlich bewirken, daß (in diesem Fall die Datei »ADRESS«) überschrieben würde ohne es zu wollen; auf Diskette vorhandene Daten wären verloren. Also eine reine Sicherheitsmaßnahme.

## Sicherheit ist Trumpf

Die Variable »SS« verhindert, daß das Programm beendet wird, ohne neu eingegebene Daten vorher abzuspeichern.

Zuletzt möchte ich noch auf einen Programmteil hinweisen, der ab Zeile 10000 steht. Dieses Unterprogramm wird immer dann angesprungen, wenn eine Datei eröffnet wird. Es können nämlich auch dann Fehler auftreten, wenn das Programm sonst einwandfrei funktioniert. Es kann zum Beispiel sein, daß sie eine schreibgeschützte Diskette benutzen wollen, um Daten abzuspeichern. Das funktioniert natürlich nicht. Damit das Programm dann nicht abstürzt, wird jedesmal nach einem Öffnen eines Floppy-Kanals diese Routine angesprungen. Sie können dann im Fehlerfall das Problem beseitigen, ohne das Programm neu starten zu müssen und damit eventuell alle eingegebenen Daten zu verlieren. Der Fehler »62« in Zeile 10062 weist übrigens darauf hin, daß ein »FILE NOT FOUND ERROR« gemeldet wurde. In diesem Fall kann eine neue Datei angelegt werden (ab 11000). Man kann natürlich auch andere Fehler direkt abfragen. Schauen Sie nach im Floppy-Handbuch von Commodore. Dort sind alle Fehlermeldungen erklärt.

Ich glaube, daß Sie viele Möglichkeiten finden werden, eine sequentielle Datei sinnvoll einzusetzen. Setzen Sie sich hin und fangen Sie an zu programmieren und zu experimentieren. Erweitern Sie Ihr Wissen durch eigene Erfahrungen. Tippen Sie nicht nur ab, seien Sie kreativ!

(Christian Schlüter/gk)

```
6120 INPUT "ORT         <CRR><CRR><CRR>.<CR
L><CRL><CRL>";OT$(ND)
6130 INPUT "TELEFON   <CRR><CRR><CRR>.<CR
L><CRL><CRL>";TE$(ND)
6140 PRINT :PRINT
6150 PRINT "W = WEITERE ADRESSE    X = EN
DE"
6160 GET R$:IF R$="" THEN 6160
6170 IF R$="W" THEN 6050
6175 SS=1:REM   DATEN NEU EINGEGEBEN
6180 RETURN
6190 :
8000 REM  ---------------------------
8010 REM  - SPEICHERN DATEI AUF DISK -
8020 REM  ---------------------------
8025 REM  L=0 WENN NOCH NICHT GELADEN
8030 IF L=0 THEN 8170
8040 CLOSE 1:OPEN 1,8,2,"@:ADRESSE,S,W"
8050 GOSUB 10000:REM   DISKETTENFEHLER?
8060 IF A1<>0 THEN 8040
8070 PRINT# 1,ND
8080 FOR I=1 TO ND
8083 :PRINT "<CLR>     SPEICHERN DER DATE
N"
8085 :PRINT I" VON "ND
8090 :PRINT# 1,NV$(I)
8100 :PRINT# 1,NN$(I)
8110 :PRINT# 1,SR$(I)
8120 :PRINT# 1,PL$(I)
8130 :PRINT# 1,OT$(I)
8140 :PRINT# 1,TE$(I)
8150 NEXT I
8160 CLOSE 1
8165 SS=2:REM  LETZTEN STAND GESPEICHERT
8170 RETURN
8180 :
9000 REM  ---------------------------
9010 REM  - LADEN DER DATEI VON DISK -
9020 REM  ---------------------------
9030 I=0
9040 CLOSE 1:OPEN 1,8,2,"ADRESSE,S,R"
9045 GOSUB 10000:REM  DISKETTENFEHLER?
9046 IF A1<>0 THEN 9030
9047 INPUT# 1,ND$:ND=VAL(ND$)
9048 I=I+1
9049 PRINT "<CLR>"I" VON "ND
9050 INPUT# 1,NV$(I)
9060 INPUT# 1,NN$(I)
9070 INPUT# 1,SR$(I)
9080 INPUT# 1,PL$(I)
9090 INPUT# 1,OT$(I)
9100 INPUT# 1,TE$(I)
9110 IF (ST AND 64)<>64 THEN 9048
9120 CLOSE 1
9130 L=1:REM   DATEI IST GELADEN
9140 RETURN
9150 :
10000 REM  ----------------------------
10010 REM  - DISKETTENFEHLER          -
10020 REM  ----------------------------
10030 PRINT "<CLR>"
10050 INPUT# 15,A1,A2$,A3,A4
10060 IF A1=0 THEN 10170
10062 IF A1=62 THEN GOSUB 11000:GOTO 101
70
10065 PRINT
10070 PRINT A1,A2$;A3;A4
10080 PRINT :PRINT
10090 PRINT "          DISKETTENFEHLER"
10100 PRINT :PRINT
10110 PRINT "       BEHEBEN SIE DEN FEHLE
"
10120 PRINT "          UND DRUECKEN SIE"
10130 PRINT
10140 PRINT "               >> F <<"
10150 GET R$:IF R$="" THEN 10150
10160 PRINT "<CLR>"
10170 RETURN
10180 :
11000 REM  --------------------------
11010 REM   - NEUE DATEI ANLEGEN
11020 REM  --------------------------
11030 :
11040 PRINT "<CLR>"
11050 PRINT "  DIE DATEI EXISTIERT NOCH
NICHT"
11060 PRINT :PRINT
11070 PRINT "N = NEUE DATEI    X = ENDE"
11080 GET R$:IF R$="" THEN 11080
11090 IF R$="X" THEN RUN
11100 IF R$<>"N" THEN 11080
11140 CLOSE 1:OPEN 1,8,2,"ADRESSE,S,W"
11150 FOR I=1 TO 7
11160 :PRINT# 1,"."
11170 NEXT I
11180 CLOSE 1
11200 RETURN

READY.
```

**Bild 2. Unsere kleine Adressendatei ist nicht ganz vollständig. Aber das ist zum Verständnis auch nicht notwendig. Beachten Sie bitte die Zeilen 131 bis 137 und 6080 bis 6130. Diese Symbole stehen anstelle der sonst so schwer zu erkennenden Steuerzeichen.**

# Castle of Doom

**Castle of Doom ist ein Adventure, das es in sich hat. Man kann nicht nur drei verschiedene Versionen auswählen, es ist zudem gar nicht einfach, bis ans Ende durchzustehen. Dazu brauchen Sie Geduld, eine gehörige Portion Phantasie und — viel Zeit.**

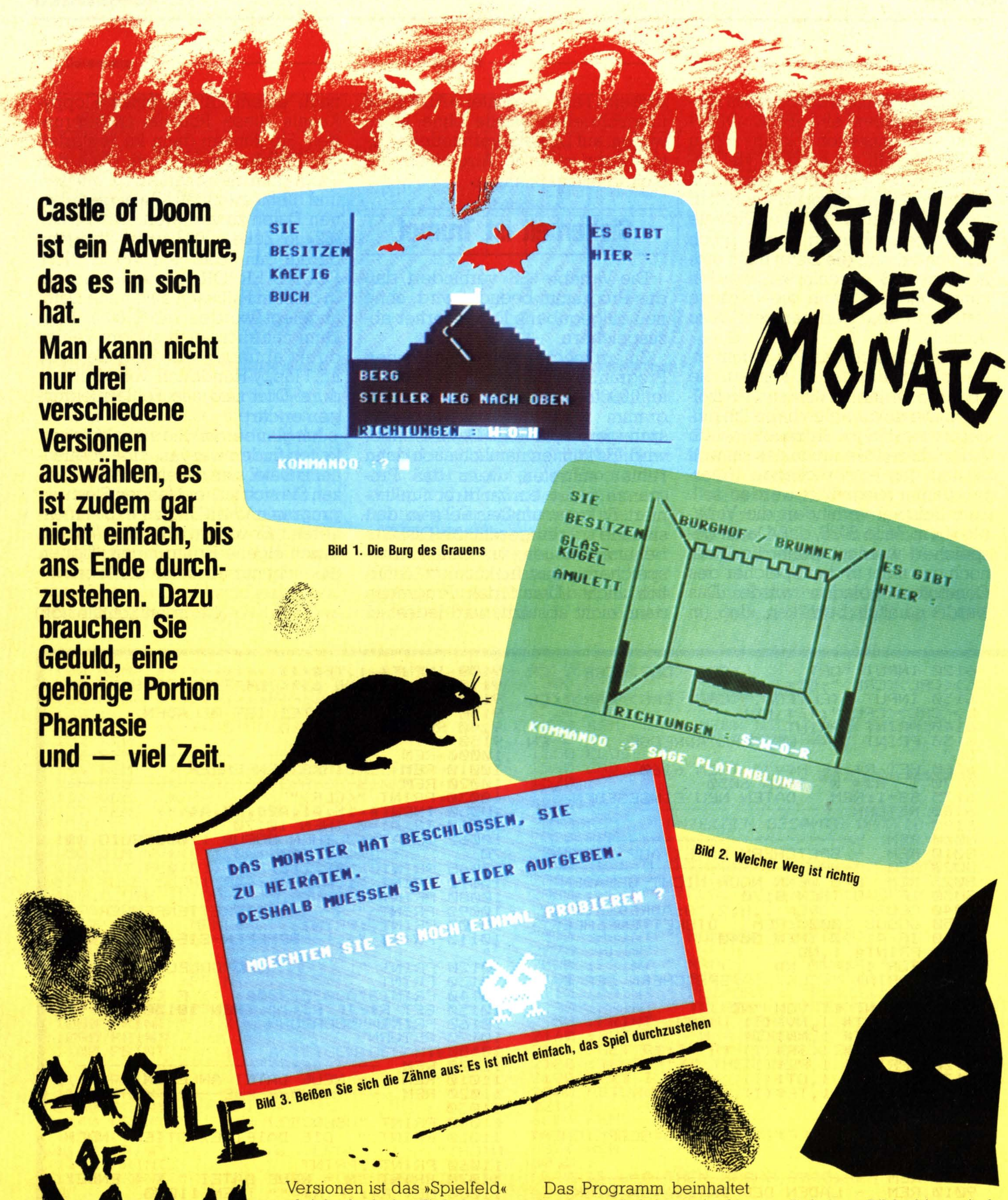



Bild 1. Die Burg des Grauens

Bild 2. Welcher Weg ist richtig

Bild 3. Beißen Sie sich die Zähne aus: Es ist nicht einfach, das Spiel durchzustehen

## CASTLE OF DOOM

Zunächst einige wesentliche Informationen zum Ablauf des Spiels:

Am Anfang des Spiels kann man sich für eine von drei verschiedenen Versionen entscheiden. Bei allen Versionen ist das »Spielfeld« gleich, doch das Ziel, das es zu erreichen gilt, ist jeweils ein anderes. Die Kommandos müssen jeweils aus zwei Worten bestehen, die durch Leerraum getrennt werden. Beide Worte können beliebig abgekürzt werden, doch man muß darauf achten, daß keine Mißverständnisse auftreten.

Das Programm beinhaltet eine kurze Anleitung, in der die wichtigsten Punkte der Befehlseingabe nochmals erwähnt werden. Zudem sind alle Verben, die der Computer versteht, aufgeführt. Wird dennoch ein Wort benutzt, das der Computer nicht kennt, so gibt er an, welches der eingegebenen Worte ihm unbekannt ist.

Auf dem Bildschirm sind immer zu sehen:

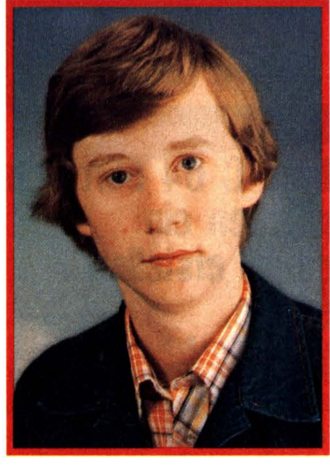
— eine Grafik, die den Ort zeigt, an dem man sich befindet,

— eine Liste der Gegenstände, die sich an diesem Ort befinden,

— eine Liste der Gegenstände, die man besitzt.

(Bernd Weißbecker/gk)

Der Gewinner des Listing des Monats



Der Autor Bernhard Weißbecker wurde am 29.10.1963 in Gelnhausen geboren. Er machte sein Abitur, und ist seit Mitte 1983 bei der Bundeswehr. Wie er zur Computerei kam, erzählt er selbst:

Meine ersten Erfahrungen mit Computern machte ich in der Schule. Dort gab es einen Computerraum, der mit einigen PETs bestückt war. Am Anfang zogen mich Spiele wie »Space Invaders« dorthin, doch bald erwarb ich auch erste Basic-Kenntnisse. Mein Beitritt zu einem Computer-Kurs, der nachmittags auf freiwilliger Basis ablief, folgte. Dieser Kurs endete leider nach einem halben Jahr schon wieder. Zudem zwang mich das nahende Abitur, meine Aktivitäten auf andere Gebiete zu verlagern.

Nun bin ich seit zehn Monaten bei der Bundeswehr und seit sieben Monaten stolzer Besitzer eines Commodore 64. Da ich mich besonders für Abenteuerspiele interessiere, wollte ich unbedingt auch selbst einmal ein solches Spiel schreiben.

An den Abenteuerspielen, die gelegentlich in Computerzeitungen abgedruckt sind, ärgerte mich immer, daß man schon beim Abtippen des Programms unfreiwillig Informationen erhält, die das Lösen des Abenteuerspiels dann oft wesentlich vereinfachen. Und mit einem Abenteuerspiel, das man einmal gelöst hat, kann man nicht mehr viel anfangen. Deshalb habe ich bei meinem Adventure drei verschiedene Versionen zur Auswahl gestellt, für die jeweils ein anderer Lösungsweg gefunden werden muß. Zudem macht es die Art der Befehlsauswertung relativ schwer, aus dem Listing zu Informationen über den Lösungsweg zu kommen.

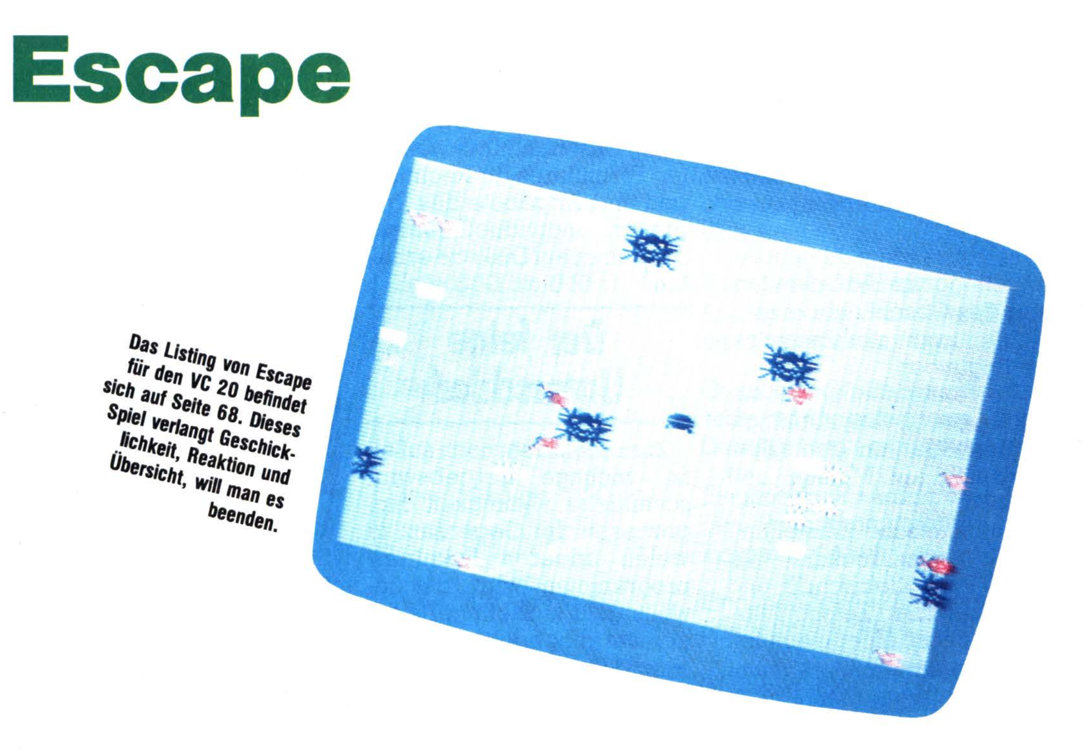
# Pac-Boy

Hier die beiden Bilder zum Listing auf Seite 85. Nicht nur die Spielfiguren wurden neu erstellt, auch die Schrift ist umdefiniert worden.

# Die Herausforderung

So sieht das Spielfeld aus. Bis zu 255 verschiedene Geschwindigkeiten können Sie Ihren Pac-Boy mit auf den Weg geben.

# Escape



Das Listing von Escape für den VC 20 befindet sich auf Seite 68. Dieses Spiel verlangt Geschicklichkeit, Reaktion und Übersicht, will man es beenden.

# Abgerechnet wird mit dem C 64

**Sie wollen Ihren C 64 beruflich nutzen und suchen ein Programm, mit dem Sie Angebote und Rechnungen betriebsindividuell erstellen können. Dann sollten Sie sich das folgende Programm näher ansehen.**

Dieses Fakturierungs- und Angebotsprogramm entspricht DIN 5001 und ist für C 64 und Epson-Drucker (mit Kreuzcodeinterface, zum Beispiel: WW Typ 9200) oder Typenradschreibmaschine Brother CE-60 beziehungsweise CE-70 geschrieben. Bei Verwendung der Schreibmaschine müssen die Steuercodes für die Druckersteuerung entsprechend der Interface-Betriebsanleitung (zum Beispiel Original-Interface IF-50 von Brother oder WDT-Interface von Wicher Digitaltechnik) abgeändert werden. Eine gedehnte Schrift ist mit einer Schreibmaschine natürlich nicht möglich.

Allgemeine Leistungen des Programmes:
- vollkommen bildschirmorientierte Programmführung
- flexibler Umsatzsteuer- und Skontosatz
- selbständige Datumseinfügung für sämtliche Rechnungen oder Angebote des Tages
- korrekte kaufmännische 5/4-Rundung
- Eingabe auch negativer Beträge (Gutschriften) mit Ausweis eines negativen Vorzeichens
- Nullen- und Dezimalpunktunterdrückung bei ganzen Stückeingaben
- Menüwahl der Arbeitsgänge und der Adressenanrede
- selbstständige Absendererstellung im Adressenfenster des Geschäftsbriefes
- Menüwahl der Zahlungsbedingungen
- Korrekturmöglichkeit vor jedem neuen Eingabeblock
- Ausgabe von deutschen Umlauten auf Monitor und Drucker
- automatische Einstellung des Epson-Druckers auf Groß- und Kleinschrift und den deutschen Zeichensatz
- Menüwahl einer eventuell gewünschten Quittierung (Barverkauf oder Nachnahme)
- automatische Formularerstellung und Formatierung, beliebige Anzahl von Kopien
- 1 Zeile (Angebot) beziehungsweise 2 Zeilen (Rechnung) frei formulierbarer Text
- betriebswirtschaftlich und steuerrechtlich korrekte Umsatzsteuer- und Skontoabrechnung
- beliebiger Wechsel zwischen den Arbeitsarten.

Leider hat dieses Programm einen kleinen Nachteil: Aufgrund der Umlautgenerierung für den Monitor (1. Block der 4 Basic-Blöcke) bleibt die Dimensionierung der maximal pro Arbeitsgang auflistbaren Artikel auf 20 beschränkt, was im Normalfall allerdings in dieser Praxis ausreichen dürfte. Eine Compilierung ist wegen des Speicherplatzangebotes (Runtime-Modul würde das Programm auf mehr als 34 Blocks vergrößern) ebenfalls nicht möglich. Wem die noch ganz akzeptable Arbeitsgeschwindigkeit (45 Sekunden für Original plus Kopie auf Epson FX 80) nicht ausreicht, hat ja die Möglichkeit, das Programm in blitzschnelles Assembler umzuschreiben. Derjenige, der das Programm nicht eintippen möchte, kann von mir auf Bestellung und Übersendung des bisher verwendeten Formulares für Rechnungen und Angebote das Programm individuell umgeschrieben auf Diskette erhalten.

## Der feine Unterschied

Zum Schluß sei noch auf eine wichtige betriebswirtschaftliche Kleinigkeit hingewiesen. Im Gegensatz zu vielen anderen Fakturierprogrammen wird die jeweils gültige Mehrwertsteuer nicht auch noch aus einem eventuell anfallenden Porto berechnet, da Postgebühren umsatzsteuerfrei sind. Ebenso bleibt bei der Skontierung das Porto unberücksichtigt, denn eine Skontierung bedingt in der Buchhaltung immer eine Umsatzsteuerkorrektur, und eine Umsatzsteuerkorrektur, die Umsatzsteuer aus umsatzsteuerfreien Leistungen korrigiert, ist bei einer Betriebsprüfung durch das Finanzamt im besten Fall peinlich. Eine langfristig viel böser ausgehende Konsequenz aus der meist falschen Skontoberechnung ist die Verfälschung der Kalkulationsfaktoren oder Handlungskostenzuschläge durch den Einfluß des Buchhaltungskontos »Kundenskonti« auf die Kostenrechnung. Folge: Aufgrund verfälschter Kalkulationsunterlagen entstehen unrealistische Verkaufspreise.

Abschließend sei noch erwähnt, daß die Umlautgenerierung bei Verwendung einer 80-Zeichen-Karte (zum Beispiel von Roos) mit festem externen Zeichensatz zwar läuft, aber keine Umlautdarstellung auf dem Monitor zuläßt. (Gerhard Schröter/rg)

## Lebenslauf



Ich bin 31 Jahre alt, Lehrer für Betriebs-, Volkswirtschaft und Englisch an der kaufmännischen Schule im Berufsschulzentrum in Biberach/Riß, seit Mitte März 1984 im Besitz eines C 64 nebst Peripherie.

Mein Weg zur EDV:

Seit März 84 Teilnahme an einer Fortbildungsveranstaltung (zwei Wochenstunden) auf der schuleigenen Anlage (Digital) und Anschaffung eines C 64 zu Übungszwecken.

Hobby: Übernachtungen vor dem Keyboard

**Zur Entstehungsgeschichte und zur Anwendung des Programms:**

Wohl so mancher Selbständige — oder besser wohl mancher Sohn eines Selbständigen — hat zu Hause einen C 64 nebst Peripherie stehen und nutzt die Anlage zu allem Möglichen, nur nicht zu beruflichen Zwecken. Die Gründe hierfür sind vielfältig: Der Sohn will für den Vater keine Rechnungen oder Angebote schreiben, weil dies ja ein glatter Mißbrauch der Anlage und seiner ohnehin knappen Freizeit wäre. Der Vater würde die Anlage schon beruflich nutzen, aber die Fakturierungsprogramme sind nicht ganz ohne jedes Verständnis für eine EDV-Anlage zu handhaben. Außerdem sind gute Fakturierprogramme auch noch teuer. Die billigen Fakturierungsprogramme dagegen sind nicht betriebsindividuell genug, entsprechen nicht der DIN 5001 (Briefschreibenorm) oder sind betriebswirtschaftlich oder steuerrechtlich schlicht unkorrekt. Fazit: Die von manchem in den Homecomputer gesetzten Hoffnungen werden aufgegeben, und Vati tippt seine Rechnungen und Angebote in zeitraubender Arbeit und unter hohem »Tip-Ex-Verbrauch« mühselig in die mit teuren, vorgedrückten Formblättern gefütterte Schreibmaschine. Daß es auch anders geht, versucht das vorgestellte Fakturierungsprogramm zu zeigen.

(Gerhard Schröter/rg)

# Programm und Erläuterung der Anwendung des Monats

Hinweise für die Listingeingabe (im folgenden sollen einige gegeben werden). In den Zeilen 280 bis 320 und in der Zeile 2250 müssen beim Eingeben des Listings Ihre individuellen Angaben eingesetzt werden. Sind in diesen Eingaben Umlaute enthalten, so müssen diese als Grafikzeichen (siehe Tastaturbelegung) programmiert werden. Verändern Sie auf keinen Fall die Dimensionierung in Zeile 240!
Zeile 240 Und 250:
§ entspricht dem Kaufmanns-a und ü der rechten Eckklammer.
Zeile 320:
Der letzte Buchstabe Ihres individuellen Geschäftsortes muß auf Höhe des »t« des alten Mustergeschäftsortes kommen.
Zeile 60400:
ß entspricht dem Potenzierungspfeil (senkrechter Pfeil).

**Tastaturbelegung:**

| | |
|---|---|
| shift / + | = ß |
| shift / linke Eckklammer | = ö |
| shift / rechte Eckklammer | = ä |
| Kaufmanns-a | = ü |
| shift / Kleinerzeichen | = Ö |
| shift / Größerzeichen | = Ä |
| shift / Kaufmanns-a | = Ü |

**Variablenliste** (in der Reihenfolge des erstmaligen Vorkommens)

| | |
|---|---|
| i | = Zählergröße |
| a | = Datagröße |
| an$ | = Artikelname |
| m | = Artikelmenge |
| p | = Artikeleinzelpreis |
| l$ | = Text (auch h$, k$, f$, ff$, fo$, fb$, bz$, g$) |
| wt$ | = Übergabe- und Rückgabevariable für Textformatieruns- und Umlautunterprogramm |
| d$ | = Datum |
| sn | = Skonto |
| j$ | = j/n-Sicherheitsabfrage |
| a | = Aufgabenart |
| x1$ | = Text (bis c6$) |
| ar | = Anredewahl (ar$=Anredeart) |
| v | = Index für Artikel |
| vp | = Verpackungskosten |
| f | = Versandkosten |
| n$ | = Empfängername |
| s$ | = Straße |
| o$ | = Ort |
| iz$ | = Bezugszeichenzeileninhalt (auch in$, uz$, un$) |
| 10$ | = Formgröße (auch 11$, 12$, 13$, 14$) |
| za | = Zahlungsartwahl |
| a$ | = Anmerkungenwahl |
| al$ | = Anmerkung (auch a2$ und a6$) |
| ak | = Kopienanzahl |
| tl | = maximale Textlänge |
| tx$ | = Übergabevariable für Text |
| vk | = Vorkommastellen |
| nk | = Nachkommastellen |
| u$ | = Vor- und Nachkommastellen |
| ux | = Übergabevariable |

**Bedienungshinweise:**

Nach RUN muß wegen der Umlautgenerierung bis zum eigentlichen Programmbeginn zirka 45 Sekunden gewartet werden. Sämtliche im einmal laufenden Programm erscheinenden j/n-Fragen können anstelle von »j« für ja auch mit der RETURN-Taste beantwortet werden; dies gilt nicht für den Eingabeblock »Anmerkungen«. Werden Anmerkungen gewünscht, so sind diese — falls sie ein Komma enthalten — in Anführungszeichen zu setzen. Die Eingabe des Tagesdatums sollte wegen der Bezugszeichen-Zeilenformatierung immer zehnstellig in Dezimalform (zum Beispiel: 01.01.1984) erfolgen. Wenn Sie eine Typenrad-Schreibmaschinen als Ausgabeeinheit verwenden, haben Sie den linken und rechten Rand von Hand einzustellen — korrigieren Sie dann auch wegen des Walzenschlupfes gelegentlich die Papierführung des Endlospapieres. Sollten Sie einmal nach dem ersten RUN kein befriedigendes Cursorzeichen erhalten, dann schalten Sie Ihren C 64 ab und laden nochmals.

| Zeilen | | Bedeutung |
|---|---|---|
| Zeile 10 bis 230 | : | Einstellung des Computers auf Groß-/Kleinschrift und Generierung der Umlaute in Block 1 der insgesamt vier Basicblöcke |
| Zeile 240 bis 350 | : | Variablendimensionierung, Definition von Standardtext mit Sprung ins Unterprogramm zur Umlautwandlung für die Druckerausgabe |
| Zeile 620 bis 670 | : | Festlegung von sich am jeweiligen Arbeitstag nicht ändernden Größen wie zum Beispiel dem Datum. Ende des »Vorprogramms« — wird bis zum neuerlichen Lagen des Programmes nicht mehr berührt |
| Zeile 680 bis 840 | : | Menüs zur Wahl der Arbeitsart, der Anrede und entsprechende Definition von Variablen |
| Zeile 850 bis 910 | : | Eingabeblock »Empfänger« mit Sprung in die Umlautwandlung und Sicherheitsabfrage |
| Zeile 920 bis 1000 | : | Eingabeblock »Bezugszeichenzeile« mit Umlautwandlung und Definition von Formstandards |
| Zeile 1010 bis 1110 | : | Eingabeblock »Artikel, Mengen, Preise« mit Variablenindizierung durch Zähler |
| Zeile 1120 bis 1210 | : | Eingabeblöcke »Zahlungsart« und »Nebenkosten« |
| Zeile 1220 bis 1295 | : | Eingabeblöcke »Anmerkungen«, aufgeteilt nach Arbeitsart |

| | | |
|---|---|---|
| Zeile 1300 bis 1315 | : | Eingabeblock »Anzahl der Kopien« |
| Zeile 1320 bis 2370 | : | Erstellung der Erstschrift mit Rechenteilen, Sprünge in Formatierungsprogramme und Standardtexten |
| Zeile 2380 bis 2490 | : | Kopierschleife und Eingabeblöcke zum weiteren Arbeitsablauf mit Rücksprunganweisung zum Hauptprogrammbeginn beziehungsweise zum Hauptprogrammende |
| Zeile 40100 | : | Unterprogramm Textformatierung |
| Zeile 45000 bis 45110 | : | Unterprogramm Umlautwandlung für Drucker |
| Zeile 50100 bis 50300 | : | Unterprogramm Zahlenformatierung mit Nullen- und Dezimalpunktunterdrückung (falls Dezimalwert erscheint, wird der Dezimalpunkt doch ausgewiesen) |
| Zeile 60100 bis 61500 | : | Unterprogramm Zahlenformatierung mit permanenter Dezimalpunkt- und Nullenausgabe |

**Programmbeschreibung nach Zeilennummern**

```
ABELE & BEBELE OHG      Textilversand
Postfach 1 23, 9870 Abcstadt 6, Tel.: 09999/1234567
Bankkonto: Kreissparkasse Abcstadt (BLZ 654 500 70) Nr. 239 699

ABELE & BEBELE OHG, Postfach 1 23, 9870 Abcstadt 6

Herr
Gerhard Schröter
Schießhüttenweg 1

7951 Ingoldingen 1

Ihre Zeichen u. Nachricht, unsere Zeichen u. Nachricht       Abcstadt
GS              15.06.84   AB/2/vk              --          03.07.1984

Angebot

Angebotsnummer:11-7-84
Angebotsdatum :03.07.1984

Sehr geehrte Damen und Herren,

aufgrund Ihrer Anfrage bieten wir Ihnen folgende Artikel an:

Artikelname od. Leistung   Einheiten   Preis/Einheit  Gesamtpreis/DM
---------------------------------------------------------------------
Winterrmantel ''Taiga''        1            510.00          510.00
Skipulli ''Arlberg''           1            123.00          123.00
Herrenhose                     1             89.00           89.00
---------------------------------------------------------------------
Summe, Nettowarenwert:                                      722.00
Verpackung/Fracht:                                            0.00
Gesamtpreis exkl. Mwst.:                                    722.00
+ 14 % gesetzl. Mwst.:                                      101.08
Versandkosten (Porti):                                        0.00
                                                     -------------
Gesamtpreis/DM: >>>>>>                                      823.08
                                                     =============

Zahlbar innerhalb 30 Tage, netto, oder innerhalb von 10 Tagen
mit 2.5 % Skontoabzug (20.58 DM).

Angebot freibleibend!

Wir bitten um Ihren Auftrag und sichern Ihnen sorgfältige Ausführung
zu.

Mit freundlichen Grüssen

ABELE & BEBELE OHG
```

**Eine Musterrechnung**

**Listing »Faktan 64« (Anfang). Die Zeile 750 ist durch folgende Statements zu ergänzen: ww$ = "":VU$=" "**

```
10 printchr$(147):printchr$(14):printchr
$(8)
15 print" * * * * * * * * * * * * * * *
* * * * "
20 print" *
    * "
25 print" *          F A K T A N  C-64
    * "
30 print" *
    * "
35 print" *    (C.) by  Gerhard Schroeter
    * "
40 print" *              Tel.:07355/1285
    * "
55 print" *
    * "
60 print" * * * * * * * * * * * * * * *
* * * * "
70 print:print:print" Guten Tag!":print:
print" Bitte warten Sie!":print:print
90 printchr$(32)chr$(91)chr$(60),chr$(93
)chr$(62),chr$(64)chr$(186),chr$(123)
95 poke52,48:poke56,48:clr:poke56334,pee
k(56334)and254:poke1,peek(1)and251
100 fori=0to1023:pokei+12288,peek(i+2048
+53248):next
110 fori=0to7:reada:poke12288+i,a:next
120 fori=0to7:reada:poke12288+122*8+i,a:
next
130 fori=0to7:reada:poke12288+27*8+i,a:n
ext
140 fori=0to7:reada:poke12288+60*8+i,a:n
ext
150 fori=0to7:poke12288+29*8+i,peek(5324
8+2048+8+i):next
160 poke12288+29*8+1,102
170 fori=0to7:reada:poke12288+62*8+i,a:n
ext
180 fori=0to7:reada:poke12288+91*8+i,a:n
ext
190 poke1,peek(1)or4:poke56334,peek(5633
4)or1
200 poke53272,(peek(53272)and240)+12
210 data0,0,102,0,102,102,60,0,102,0,102
,102,102,102,60,0
220 data0,0,102,60,102,102,60,0,102,60,1
02,102,102,102,60,0
230 data102,0,60,102,126,102,102,0,0,56,
108,120,108,108,120,96
240 diman$(20):dimm(20):dimp(20):l$="zu.
":h$="Mit freundlichen Gr`ssen"
250 k$="Wir bitten um Ihren Auftrag und
sichern Ihnen sorgfültige Ausf`hrung"
260 wt$=k$:gosub45000:k$=wt$:wt$=h$:gosu
b45000:h$=wt$
280 f$="ABELE & BEBELE OHG   Textilversa
nd":g$="ABELE & BEBELE OHG"
290 ff$="ABELE & BEBELE OHG, Postfach 1
23, 9870 Abcstadt 6"
300 fo$="Postfach 1 23, 9870 Abcstadt 6,
 Tel.: 09999/1234567"
310 fb$="Bankkonto: Kreissparkasse Abcst
adt (BLZ 654 500 70) Nr. 239 699"
320 bz$="Ihre Zeichen u. Nachricht, unse
re Zeichen u. Nachricht      Abcstadt"
330 wt$=f$:gosub45000:f$=wt$:wt$=g$:gosu
b45000:g$=wt$
340 wt$=ff$:gosub45000:ff$=wt$:wt$=fo$:g
```

```
osub45000:fo$=wt$
350 wt$=fb$:gosub45000:fb$=wt$:wt$=bz$:g
osub45000:bz$=wt$:printchr$(147)
620 input" Heutiges Datum     : ";d$:prin
t:input" Mehrwertsteuersatz: ";t:print
640 input" Skontosatz         : ";sn:prin
t:print:j$="j"
650 input" Alles richtig (j/n) : ";j$:if
j$=""orj$="j"then680
670 ifj$="n"thenprintchr$(147):goto620
680 printchr$(147):print" Aufgabe: Rechn
ung schreiben    =  1":a=0
690 print:print"              Angebot schrei
ben      =  2":print
700 input" Aufgabenart    1  oder  2
 : ";a:print:ifa<1ora>2then680
712 ifa=1thenx1$="Rechnung":x2$="Rechnun
gsnummer:":x3$="Rechnungsdatum :"
720 ifa=2thenx1$="Angebot":x2$="Angebots
nummer:":x3$="Angebotsdatum :"
730 ifa=2thenx4$="Sehr geehrte Damen und
 Herren,"
735 ifa=2thenx5$="aufgrund Ihrer Anfrage
 bieten wir Ihnen folgende Artikel an:"
740 printchr$(147):n$="":s$="":o$="":nr$
="":iz$="":in$="":uz$="":un$="":v=0
750 vp=0:f=0
770 print" Anrede:    Firma        = 1":p
rint"            Herr         = 2"
790 print"            Frau         = 3":p
rint
800 ar=0:input" Anredewahl  1 bis 3    :"
;ar:ifar<1orar>3then740
820 ifar=1thenar$="Firma"
830 ifar=2thenar$="Herr"
840 ifar=3thenar$="Frau"
850 print:print:input" Empfaengername :
";n$:print:wt$=n$:gosub45000:n$=wt$
860 input" Strasse u. Nr. : ";s$:print:w
t$=s$:gosub45000:s$=wt$
870 input" PLZ und Ort     : ";o$:print:w
t$=o$:gosub45000:o$=wt$
875 ifa=2theninput" Angebotsnummer : ";n
r$:print:print:goto890
880 input" Rechnungsnummer: ";nr$:print:
print
890 j$="j":input" Alles richtig (j/n) :
";j$:ifj$=""orj$="j"then920
910 ifj$="n"thenprintchr$(147):goto740
920 printchr$(147):input" Ihre Zeichen
     : ";iz$:print
925 wt$=iz$:gosub45000:iz$=wt$
930 input" Ihre Nachricht      : ";in$:pr
int:wt$=in$:gosub45000:in$=wt$
940 input" unsere Zeichen      : ";uz$:pr
int:wt$=uz$:gosub45000:uz$=wt$
950 input" unsere Nachricht    : ";un$:pr
int:print:wt$=un$:gosub45000:un$=wt$
960 j$="j":input" Alles richtig (j/n) :
";j$:ifj$=""orj$="j"then980
970 ifj$="n"then920
980 l0$="Artikelname od. Leistung   Einh
eiten   Preis/Einheit  Gesamtpreis/DM"
985 l1$="--------------------------------
--------------------------------------"
990 l2$="--------------------------------
--------------------------------------"
995 l3$="                  ---------------"
1000 l4$="
                  ============="
1010 v=v+1
1015 an$="":m=0:p=0:printchr$(147):input
" Artikel/Leistung : ";an$:print
1020 wt$=an$:gosub45000:an$=wt$
1030 input" Menge              : ";m:print
1040 input" Einzelpreis        : ";p:print
:print
1045 an$(v)=an$:m(v)=m:p(v)=p:j$="j":inp
ut" Alles richtig  (j/n) : ";j$:print
1060 ifj$=""orj$="j"then1090
1070 ifj$="n"then1015
1090 j$="":print:input" Weitere Artikel
(j/n) : ";j$:print
1100 ifj$=""orj$="j"then1010
1110 ifj$="n"thenprintchr$(147):print
1120 input" Verpackungskosten/Fracht : "
;vp:print
1130 input" Versandkosten (Porti)    : "
;f:print
1132 j$="j":input" Alles richtig  (j/n)
: ";j$:print:ifj$=""orj$="j"then1135
1133 ifj$="n"thenprintchr$(147):goto1120
1135 ifa=2thenza=2:goto1220
1140 printchr$(147):print" Zahlungsart:
Barverkauf = 1"
1150 print"               Ziel       = 2"
1160 print"               Nachnahme  = 3"
1170 za=0:print:input" Zahlungsart   1 b
is 3 : ";za:print:print
1180 ifza<1orza>3thenprintchr$(147):prin
t:goto1140
1190 j$="j":input" Alles richtig  ( j /
n )  : ";j$:print
1200 ifj$="j"orj$=""then1220
1210 ifj$="n"thenprintchr$(147):print:go
to1140
1220 ifa=2then1270
1225 printchr$(147):print:a$="n":input"
Anmerkungen  (j/n) : ";a$:print
1228 a1$="":a2$=""
1230 ifa$="j"thenprint:goto1240
1235 ifa$=""ora$="n"then1300
1240 print" ACHTUNG max. je der 2 Doppel
zeilen bis  zum Sternchen schreiben! *"
1245 print:inputa1$:print:inputa2$:print
1247 wt$=a1$:gosub45000:a1$=wt$:wt$=a2$:
gosub45000:a2$=wt$
1250 i$="j":input" Alles richtig (j/n) :
";i$
1255 ifi$=""ori$="j"then1300
1260 ifi$="n"then1225
1270 printchr$(147):print:a$="n":input"
Anmerkung (j/n) : ";a$:print:a6$=""
1275 ifa$="j"thenprint:goto1285
1280 ifa$=""ora$="n"then1300
1285 print" ACHTUNG in der Doppelzeile n
ur bis zum  Sternchen schreiben!      *"
1290 print:inputa6$:wt$=a6$:gosub45000:a
6$=wt$:i$=""
1291 print:input" Alles richtig (j/n) :
";i$
1292 ifi$=""ori$="j"then1300
1295 ifi$="n"then1270
1300 printchr$(147):print:print
1310 input" Anzahl der Kopien : ";ak:pri
nt
1315 print:print" Bitte warten Sie !"
1320 open1,4,7
1330 print#1,chr$(27);chr$(114);chr$(2);
:print#1,chr$(27);chr$(76);chr$(9);:s=0
```

**Listing »Faktan 64« (Fortsetzung)**

```
1390 print#1,chr$(27);chr$(119);chr$(1);
:print#1,f$
1410 print#1,chr$(27);chr$(119);chr$(0);
:print#1,fo$:print#1,fb$
1480 print#1,chr$(27);chr$(33);chr$(4);:
print#1,chr$(27);chr$(45);chr$(1);
1500 print#1:print#1:print#1:print#1,ff$
1510 print#1,chr$(27);chr$(45);chr$(0);:
print#1,chr$(27);chr$(33);chr$(0);
1530 print#1:print#1:print#1,ar$
1580 print#1,n$:print#1,s$:print#1:print
#1,o$:print#1:print#1:print#1
1590 print#1:print#1,bz$:tl=16:tx$=iz$:g
osub40100:tl=11:tx$=in$:gosub40100
1670 tl=18:tx$=uz$:gosub40100:tl=13:tx$=
un$:gosub40100:tl=11:tx$=d$:gosub40100
1710 print#1:print#1:print#1,chr$(27);ch
r$(119);chr$(1);:print#1,x1$
1720 print#1,chr$(27);chr$(119);chr$(0):
print#1,x2$;nr$:print#1,x3$;d$
1730 ifa=2thenprint#1:print#1:print#1,x4
$:print#1:print#1,x5$:print#1:goto1770
1740 print#1:print#1:print#1,"Wir liefer
ten Ihnen:"
1770 print#1:print#1,l0$:print#1,l1$:tl=
26:vk=6:nk=2:u$="13.2":n=(v)-kz
1800 forv=1ton:tx$=an$(v):gosub40100
1830 zx=m(v):gosub50100:ux=p(v):gosub601
00:g=m(v)*p(v):ux=g:gosub60100:s=s+g
1880 print#1,:nextv:ifv=nthen1900
1900 print#1,l2$:tl=30:u$="35.2":tx$="Su
mme, Nettowarenwert:":gosub40100
1930 ux=s:gosub60100:print#1,:tx$="Verpa
ckung/Fracht:":gosub40100
1960 ux=vp:gosub60100:print#1,:gp=s+vp:t
x$="Gesamtpreis exkl. Mwst.:"
2000 gosub40100:ux=gp:gosub60100:print#1
,:mw=(gp*t)/100
2030 t2$=str$(t):t1$="+"+t2$+" % gesetzl
. Mwst.:":tx$=t1$:gosub40100:ux=mw
2050 gosub60100:print#1,:tx$="Versandkos
ten (Porti):":gosub40100
2055 ux=f:gosub60100
2090 print#1,:print#1,l3$:ge=gp+mw+f:tx$
="Gesamtpreis/DM: >>>>>>":gosub40100
2130 ux=ge:gosub60100:print#1,:print#1,l
4$:print#1:print#1
2160 sk=(gp+mw)*(sn/100):so=int(sk*100+0
.5)/100:ifso=0thenso$="00"
2180 ifso>1thenso$=str$(so):so$=right$(s
o$,len(so$)-1)
2190 ifso<1thenso$=str$(so):so$="0"+righ
t$(so$,len(so$)-1)
2192 ifa=1then2200
2195 ifa=2then2260
2200 ifza=0then2350
2210 ifza=1then2240
2220 ifza=2then2260
2230 ifza=3then2280
2240 print#1,"Betrag erhalten.":print#1
2250 print#1,"Abcstadt, den ";d$;"
........................":goto2290
2260 print#1,"Zahlbar innerhalb 30 Tage,
 netto, oder innerhalb von 10 Tagen"
2270 print#1,"mit";sn;"% Skontoabzug (";
so$;" DM).":goto2290
2280 print#1,"Betrag durch Nachnahme erh
alten.":goto2290
2290 ifa=1then2300
2294 ifa$="n"then2340
2296 ifa$="j"then2335
2300 ifa$="n"then2350
2320 print#1:print#1,a1$:print#1,a2$:got
o2350
2335 print#1:print#1,a6$
2340 print#1:print#1,k$:print#1,l$:print
#1:print#1,h$:print#1:print#1,g$
2350 print#1,chr$(12);
2360 ifak>0then2380
2370 ifak=0then2424
2380 forc=1toak:kz=x+1:ak=ak-1:gp=0:goto
1330:nextc
2424 ifa=1then2430
2425 ifa=2thenprintchr$(147):print:input
" Weitere Angebote  (j/n) : ";wa$:print
2426 ifwa$=""orwa$="j"thenclose1:goto740
2427 ifwa$="n"then2460
2430 printchr$(147):print:input" Weitere
 Rechnungen (j/n) : ";wr$:print
2440 ifwr$="j"orwr$=""thenclose1:goto740
2450 ifwr$="n"then2460
2460 print#1,chr$(27);chr$(64);:close1
2470 print:print:input" Weitere Arbeiten
  (j/n) : ";j$:ifj$=""orj$="j"then680
2490 ifj$="n"thenprintchr$(147):end
40100 tx=len(tx$):print#1,tx$;tab(tl-tx)
;:return
45000 forl=1tolen(wt$):vu$=mid$(wt$,l,1)
45020 ifvu$=chr$(219)thenvu$=chr$(94)
45030 ifvu$=chr$(64)thenvu$=chr$(93):got
o45090
45040 ifvu$=chr$(186)thenvu$=chr$(125)
45050 ifvu$=chr$(91)thenvu$=chr$(92)
45060 ifvu$=chr$(60)thenvu$=chr$(124)
45070 ifvu$=chr$(93)thenvu$=chr$(91)
45080 ifvu$=chr$(62)thenvu$=chr$(123)
45090 ww$=ww$+vu$
45100 ifvu$=chr$(93)thenvu$="":goto45040
45110 nextl:wt$=ww$:ww$="":return
50100 zx=int(zx*100+0.5)/100:zx$=str$(zx
)
50200 zl=len(zx$):zu=0:forzn=1tozl:ifmid
$(zx$,zn,1)="."goto50400
50300 zu=zu+1:nextzn
50400 print#1,tab(vk-zu);zx;tab((nk+vk)-
(vk-zu+zl)+1);:return
60100 uv$=right$(u$,1):ul=int(val(u$)):i
fvu$<>"."thenur=val(uv$):goto60400
60300 ua$=str$(sgn(ux)*int(abs(ux)))+"."
:ub$="":ul=ul+1:goto 61300
60400 ul=int(val(u$)):ux$=str$(sgn(ux)*(
int(abs(ux)*10ßur+.5))/10ßur)
60600 uv=0:forun=1tolen(ux$):ifmid$(ux$,
un,1)="."thenuv=un
60650 nextun
60700 if uv=0 then uv=un:ux$=ux$+"."
60900 if uv<>2 then 61100
61000 ux$=left$(ux$,1)+"0"+right$(ux$,le
n(ux$)-1):ul=ul-1:ur=ur+1
61100 ub$=mid$(ux$,uv,len(ux$)+1)+"00000
0000"
61200 ub$=left$(ub$,ur+1):ua$=left$(ux$,
uv-1)
61300 if len(ua$)>ul then print"usingber
eich zu klein":stop
61400 if len(ua$)<ul then ua$=" "+ua$:go
to 61400
61500 print#1,(ua$+ub$);:return
ready.
```

**Listing »Faktan 64« (Schluß)**

Anmerkung: Da die Unterprogramme mit Übergabevariablen aus dem Hauptprogramm (zum Beispiel wt$ oder tx$) versorgt werden und die Arbeitsvariablen der Unterprogramme, wie zum Beispiel tl (Textlänge), vk (Vorkommastellen), nk (Nachkommastellen) oder u$ (Vor- und Nachkommastellen bei Dezimalpunkt- und Nullenausgabe), im Hauptprogramm (siehe zum Beispiel Zeile 1770) beliebig definiert werden können, lassen sich diese als Bausteine für andere Programme nutzen.

**Änderungen bei Verwendung einer Typenradschreibmaschine**
Brother CE-60/CE-70,
Privileg P 6000,
Schweiz: Migro-Office,
auch: CE-50 m. IF-An.,
ebenso: Schönschreibdrucker Brother EM-80/100/200

Bei Verwendung eines Schönschreibdruckers beziehungsweise einer Typenradmaschine von Brother zusammen mit dem WDT-Interface (Wicher Digital Technik, Flörsbachtal, baugleich mit Quelle und Neckermann) sind folgende Änderungen im Listing notwendig:

```
1310 input" Anzahl der Kopien : ";ak:print
1311 print:print" Bitte warten Sie !"
1312 bl=30:ifa=2anda$="j"thenbl=18
1313 ifa=2anda$="n"thenbl=20
1314 ifa=2thenax=2:goto1319
1315 ifza=1thenht=3
1316 ifza=2thenht=2
1317 ifza=3thenht=1
1318 ax=0:ifa$="j"thenax=3:goto1319
1319 z=bl-ht-ax-(v)
1320 open1,4,7
1330 s=0:print#1,f$:print#1,fo$:print#1,fb$
1480 print#1,chr$(30);:print#1,chr$(6);
1500 print#1:print#1:print#1:print#1,ff$
1510 print#1,chr$(25);:print#1,chr$(7);
1530 print#1:print#1:print#1,ar$
ready.
1710 print#1:print#1:print#1,x1$:print#1
1720 print#1,x2$;nr$:print#1,x3$;d$
ready.
2350 forq=1toz:print#1:nextq
 ready.
1480 print#1,chr$(27);chr$(31);chr$(9);
:print#1,chr$(27);chr$(69);
ready.
1510 print#1,chr$(27);chr$(31);chr$(13):
:print#1,chr$(27);chr$(82);
ready.
```

**Automatischer Seitenvorschub, komprimierte Schrift und automatische Unterstreichung**

```
45000 forl=1tolen(wt$)
45010 vu$=mid$(wt$,1,1)
45020 ifvu$=chr$(219)thenvu$=chr$(190)
45030 ifvu$=chr$(64)thenvu$=chr$(189)
45040 ifvu$=chr$(186)thenvu$=chr$(221)
45050 ifvu$=chr$(91)thenvu$=chr$(188)
45060 ifvu$=chr$(60)thenvu$=chr$(220)
45070 ifvu$=chr$(93)thenvu$=chr$(187)
45080 ifvu$=chr$(62)thenvu$=chr$(219)
45090 ww$=ww$+vu$
45100 nextl:wt$=ww$:ww$="":return
(45110 entfällt)
```

**Zusatzänderung bei Verwendung des Original-Interface Brother IF-50**

# Kopplung über den User-Port

**In der letzten Ausgabe brachten wir zwei VC 20 dazu, gegeneinander zu spielen. Nun lassen wir zwei C 64, oder einen VC 20 und einen C 64, Daten ohne jegliches Interface über den User-Port austauschen.**

Seit einigen Monaten betreibe ich zwei gekoppelte C 64, die sich zu meiner vollen Zufriedenheit die Befehle eines fünfstimmigen Synthesizer-Programms teilen. Während es für meine speziellen Anforderungen bereits genügt, ein einzelnes Byte rasch zu transportieren, können die vorhandenen Routinen ohne viel Mühe so erweitert werden, daß man

a) Basicprogramme
b) Variablen,
c) Arrays und
d) Maschinenprogramme überträgt und gegebenenfalls ausführt.

In diesem Artikel erläutere ich Ihnen die Grundroutinen anhand der Basicprogrammübertragung.

Neben der eingebauten seriellen Schnittstelle hat der C 64 wie alle Computer von Commodore einen programmierbaren User-Port, der eine Verbindung nach »draußen« darstellt und entsprechend genutzt werden kann. Die Adresse des User-Ports liegt beim C 64 auf 56577 ($dd01). Eine Abfrage mit PEEK (56577) zeigt den Wert 255. Ein POKE-Befehl ändert daran nichts, weil das Betriebssystem beim Initialisieren den User-Port als Eingang definiert und sich unbeschaltete Eingänge als logisch »1« verhalten. Bevor man den User-Port als Ausgang definiert, müssen einige Sicherheitsvorkehrungen getroffen werden, denn der User-Port des einen Computers wird Draht für Draht mit dem des anderen verbunden. Dabei darf immer nur ein Computer den User-Port als Ausgang betreiben, sonst kommt es zur Zerstörung der Port-Bausteine. Die Adresse des Richtungsregisters ist 56579 ($dd03). In diesem Register definiert ein gesetztes Bit das entsprechende User-Bit als Ausgang. Dadurch wird ein gemischter Betrieb möglich, der in meinem Programm jedoch nicht vorkommt.

Als Voraussetzung muß eine physikalische Verbindung in Form von Drähten und zwei User-Port-Steckern geschaffen werden.

Es sind insgesamt 11 Drähte, welche die beiden Computer verbinden. Ich habe die Verbindung mit einer 4 m langen, sorgfältig verlöteten Flachleitung realisiert und bisher keine Störung bemerkt.

Auf Seite 143 des C 64 Handbuchs ist die Kontaktbelegung des User-Port-Steckers aufgeführt.

Die Daten-Bit-Kontakte PB0 bis PB7 liegen genau in der Mitte des Steckers und werden »Bit für Bit« verbunden, also »C« an »C«, »D« an »D« und so weiter.

Auch die Masse-Leitungen »A« werden miteinander verbunden.

Die beiden übrigen Kontakte »B« und »M« werden gekreuzt, so daß »B« mit »M« verbunden ist und umgekehrt.

Das Programm ist ausführlich erklärt. Eine Besonderheit ist die Speicherbereichswahl. Jeder Autor glaubt, der $c-Block sei ganz frei und warte auf genau sein Programm.

Um hier eine bequeme Übernahmemöglichkeit in eigene Routinen zu bieten, lädt sich der Maschinenteil des Programms an das obere verfügbare RAM-Ende, schützt das Segment vor den Strings und löscht sich letztlich selbst. Die SYS-Adressen werden auf dem Bildschirm angezeigt. Dann können im Direkt-Modus Basicprogramme hin- und hergeschoben werden, oder aus Maschinenroutinen einzelne Bytes transportiert werden. Die Basicprogramm-Übertragung hat die angenehme Eigenschaft, das transportierte Programm an das bereits im Speicher befindliche Programm anzuhängen, es handelt sich also um eine Art Merge-Funktion. Der »NEW«-Befehl schafft wieder leere Verhältnisse, wenn's mal zu voll geworden ist.

## Programmerklärung

Die Punkte »..« im Basicprogramm sind Platzhalter für die $-Seite, in die das Maschinenprogramm geladen wurde. Im folgenden gehe ich von Seite $ 7f aus. $7f00 = sys 32512 überträgt ein Basicprogramm vom Sender zum Empfänger, wo es an das im Speicher befindliche Programm angehängt wird. Zunächst wird der User-Port des Senders als Ausgang definiert. Dann wird ein Zeiger in $58 eingerichtet, der von Basic-Start ($2b) bis Basic-Ende ($2d) alle Bytes der Byte-Sende-Routine mit »jsr 7f5e« übergibt. Nach Übertragung des letzten Bytes wird der User-Port wieder als Eingang geschaltet (= Sicherheitsfall).

Mit sys 32551 ($7f27) wird ein Basicprogramm empfangen. Durch Subtraktion von 2 vom Basic-Zeiger wird die Koppeladresse gewonnen, ab der die empfangenen Bytes gespeichert werden sollen. Die Byte-Empfangs-Routine steht ab $7f7c. Der Empfänger erkennt das Basicprogramm-Ende an drei folgenden Nullen ($#00), wonach er den Empfang abbricht, durch einen CLR-Befehl ($a660) die Basic-Zeiger neu setzt, durch $a533 die Programmzeilen neu bindet und schließlich in die Basicwarteschleife nach $e385 springt.

$7f5e ist die Byte-Sende-Routine. Der Akkumulator des Prozessors wird in das Datenregister des User-Ports geschrieben, anschließend wird das dritte Bit in Register $dd00 auf »0« gesetzt. Nun bleibt das fünfte Bit in Register $dd0d des anderen Computers solange »1«, bis es gelesen wird. Durch den Lesevorgang wird es gleichzeitig gelöscht. Diesen Mechanismus kann man einfach zur Übertragungskontrolle nutzen. Das dritte Bit von $dd00 ist die VALID-Flagge. Wenn sie gesetzt ist, sind die Daten gültig. Damit der Sender weiß, wann er das Byte übertragen ansehen kann, sendet der Empfänger ein QUITTUNGs-Signal. Dieses kommt im fünften Bit des Registers $dd0d beim Sender an. Auch hier wird durch das Lesen gleichzeitig gelöscht. Braucht der Empfänger zu lange, um das Byte anzunehmen, zum Beispiel dann, wenn er gar nicht eingeschaltet ist, bricht der Sender nach zirka 1,8 Millisekunden Wartezeit den Sendevorgang ab und schaltet den USER-Port wieder auf Eingangs-Modus. Dadurch soll verhindert werden, daß die beiden Computer gleichzeitig senden, was der Hardware schaden würde. Nach erfolgter Übertragung wird das dritte Bit in $dd00 wieder auf »1« gesetzt, dabei ändert sich in $dd0d des anderen Computers nichts, es ist halt notwendig, um es wieder auf »0« setzen zu können.

$7f7c ist die Byte-Empfangs-Routine. Hier wartet der Empfänger solange in der VALID-Schleife, bis die Daten gültig sind. Wird nichts gesendet, so wartet der Empfänger fortwährend und merkt nichts. Zum Abbruch muß man dann die RUN/STOP- und die RESTORE-Taste gleichzeitig drücken. Sind die Daten gültig, so werden sie in den Akkumulator des Prozessors geladen. Der Empfänger quittiert durch einmaliges Nullen des dritten Bits von $dd00.

$7f8f definiert den User-Port als Eingang. $7f92 definiert den User-Port als Ausgang. Der Trick mit »BIT mmnn« fand auch hier Anwendung. Man spart immerhin ein Byte.

$7f9d initialisiert den User-Port. Zwar setzt das Betriebssystem den User-Port als Eingang, nimmt aber nicht die Flag in $dd0d weg, wodurch es direkt nach dem Einschalten zu einer VALID-Meldung kommt, die eine richtige Übertragung verhindert. Die VALID-Schleife in $7f9d liest so lange VALID-Flags, bis keine mehr kommen, und springt dann zur Eingangs-Richtungs-Routine.

## Praktische Ausführung

— Verbindungsleitung löten (1:1).

— Programm erfassen, SAVEn und RUNnen.

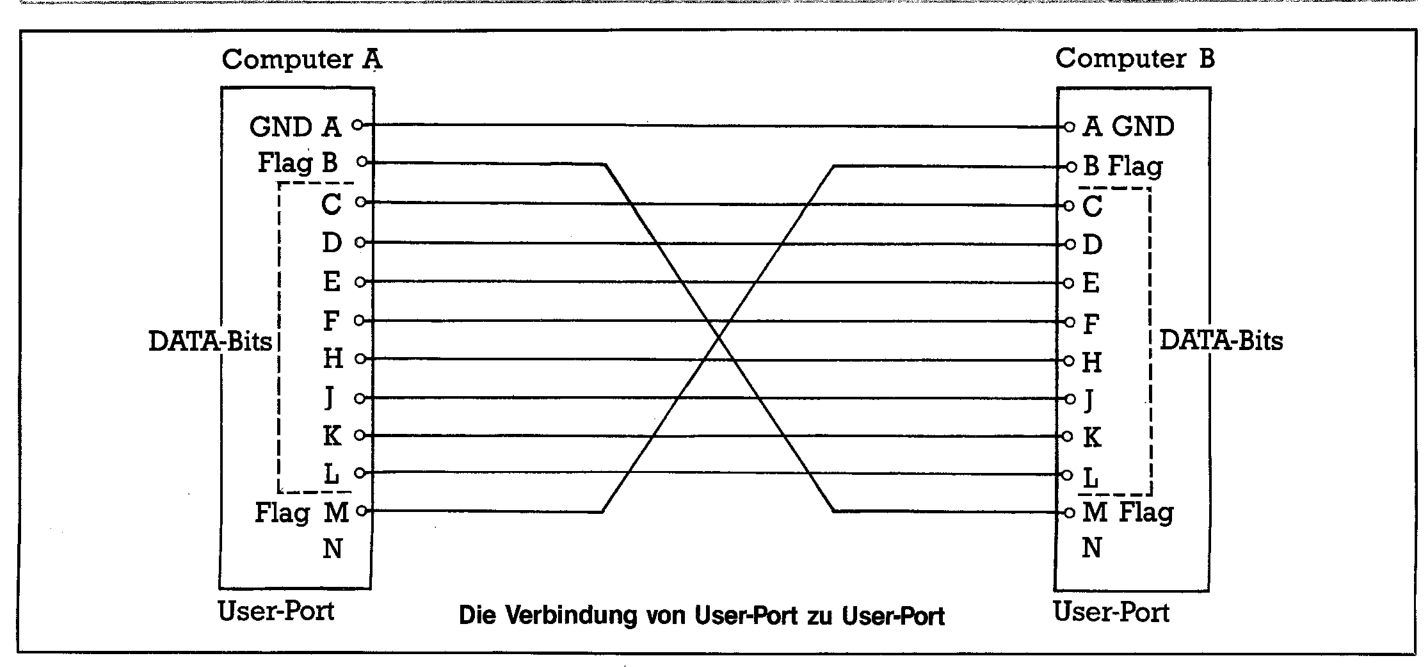
Soll zum Beispiel von Computer A ein Programm zu Computer B übertragen werden, so muß zuerst Computer B mit »sys 32551« vorbereitet werden, dann wird Computer A mit »sys 32512« zum Senden veranlaßt. Im umgekehrten Fall beschwert sich Computer A mit der Meldung »device not present error«, welche aus dem Basicinterpreter für diesen Zweck entliehen wurde. Die Übertragungsrate liegt bei etwa 11 000 Bytes pro Sekunde.

(Johannes Mockenhaupt/rg)

```
10 rem " Johannes Mockenhaupt
20 rem " Hochstadenstr. 28
30 rem "
40 rem " D-5000 Köln 1
                        ,den 29.März 1984
50 :
60 rem " C 64 / VC 20 / Koppelung
70 rem " über den programmierbaren USER
Port.
80 :
100 bytes=165:                   rem " Anz
ahl Bytes Maschinen-PRG
110 hs=peek(56)-1:               rem " hoe
chste aktuelle RAM-Seite minus 1.
120 if peek(55)>0 then hs=hs-1:rem " kei
n 'glattes' RAM-Ende: Sicherheits-Seite
130 :
140 print "▒sys"hs*256":BASIC PRG senden
.
```

**Listing zur Übertragung von Basicprogrammen zwischen zwei C 64.**

```
150 print "▒sys"hs*256+39":BASIC PRG emp
fangen.
160 print "▒sys"hs*256+94":1 BYTE senden
170 print "▒sys"hs*256+124":1 BYTE empfa
ngen
180 print "▒sys"hs*256+143":USER-Port au
f Eingang
190 print "▒sys"hs*256+146":USER-Port au
f Ausgang
200 print "▒sys"hs*256+157":QUICK-Port I
nitialisierung
290 :
300  for i=hs*256 to i+bytes-1
310   read a
320   b=a
330    if a<0 then a=a+hs+1
340   poke i,a
350   s=s+b
360  next i
370 s=s-20510
380 if s<>0 then print "▒▒Data-Fehler▒:"
s"betraegt die Differenz.":end
390 :
400 print "▒▒QuickPort▒ ist jetzt initia
lisiert.
410 poke 56,hs:     rem " neue RAM-top Ad
resse
420 sys hs*256+157:rem " Initialisierung
430 new:            rem " Setzen der BASI
C-Zeiger auf neuen Bereich.
440 :
6000 rem " Transport-Routinen      BASIC-
PRG  Senden:      $..00
6010 :
6100 data  32,146, -1:rem " ..00  20 92
..  jsr ..92    USER-Port auf Ausgang
6110 data 165, 43:     rem " ..03  a5 2b
    lda 2b        L-Byte Basic-Start
6120 data 164, 44:     rem " ..05  a4 2c
    ldy 2c        H-Byte Basic-Start
6130 data 133, 88:     rem " ..07  85 58
    sta 58          merken
6140 data 132, 89:     rem " ..09  84 59
    sty 59          merken
6200 data 160,  0:     rem " ..0b  a0 00
    ldy#00      Y-Register vorbereiten.
6210 data 177, 88:     rem " ..0d  b1 58
    lda (58).y   1 Byte aus PRG holen
6220 data  32, 94, -1:rem " ..0f  20 5e
..  jsr .. 5e   und senden.
6230 data 230, 88:     rem " ..12  e6 58
    inc 58      Zeiger erhoehen L-Byte
6240 data 208,  2:     rem " ..14  d0 02
    bne ..18    ohne Uebertrag
6250 data 230, 89:     rem " ..16  e6 59
    inc 59      Zeiger erhoehen H-Byte
6300 data 165, 89:     rem " ..18  a5 59
    lda 59      Zeiger      H-Byte
6310 data 197, 46:     rem " ..1a  c5 2e
    cmp 2e      BASIC-Ende H-Byte
6320 data 144,237:     rem " ..1c  90 ed
    bcc ..0b    noch nicht erreicht.
6330 data 165, 88:     rem " ..1e  a5 58
    lda 58      Zeiger      L-Byte
6340 data 197, 45:     rem " ..20  c5 2d
    cmp 2d      BASIC-Ende L-Byte
6350 data 144,231:     rem " ..22  90 e7
    bcc ..0b    noch nicht erreicht.
6360 data  76,143, -1:rem " ..24  4c 8f
..  jmp ..8f    USER-Port auf Eingang
6370 :
7000 rem " Transport-Routinen      BASIC-
PRG  Empfangen: $..27
7010 :
7100 data  56:          rem " ..27  38
    sec         Start-Adresse berechnen
7110 data 165, 45:     rem " ..28  a5 2d
    lda 2d        L-Byte Basic-Ende
7120 data 233,  2:     rem " ..2a  e9 02
    sbc#02        $02 substrahieren
```




Die Verbindung von User-Port zu User-Port



**Listing zur Übertragung von Basicprogrammen zwischen zwei C 64 (Schluß)**

```
7130 data 133, 45:     rem " ..2c  85 2d
    sta 2d        und speichern.
7140 data 165, 46:     rem " ..2e  a5 2e
    lda 2e        H-Byte Basic-Ende
7150 data 233,  0:     rem " ..30  e9 00
    sbc#00        Carry-Flag subtrahieren
7160 data 133, 46:     rem " ..32  85 2e
    sta 2e        und speichern.
7200 data 160,  0:     rem " ..34  a0 00
    ldy#00        Y-Register vorbereiten.
7210 data  32,124, -1:rem " ..36  20 7c
..  jsr ..7c      1 Byte empfangen
7220 data 145, 45:     rem " ..39  91 2d
    sta (2d).y    und speichern.
7230 data 230, 45:     rem " ..3b  e6 2d
    inc 2d        BASIC-PRG Ende erhoehen
7240 data 208,  2:     rem " ..3d  d0 02
    bne ..41      ohne Uebertrag
7250 data 230, 46:     rem " ..3f  e6 2e
    inc 2e        H-Byte erhoehen
7300 data 168:         rem " ..41  a8
    tay           Ende-Bedingung pruefen
7310 data 208,  8:     rem " ..42  d0 08
    bne ..4c      kein Ende
7320 data 165,252:     rem " ..44  a5 fc
    lda fc        2.Byte=0 ?
7330 data 208,  4:     rem " ..46  d0 04
    bne ..4c      nein, kein Ende
7340 data 165,253:     rem " ..48  a5 fd
    lda fd        3.Byte=0 ?
7350 data 240,  9:     rem " ..4a  f0 09
    beq ..55      ja, dann Ende
7360 data 165,252:     rem " ..4c  a5 fc
    lda fc        vorletztes Byte kellern
7370 data 133,253:     rem " ..4e  85 fd
    sta fd
7380 data 132,252:     rem " ..50  84 fc
    sty fc        letztes   Byte kellern
7390 data  24:         rem " ..52  18
    clc           Sprungbedingung
7400 data 144,223:     rem " ..53  90 d9
    bcc ..34      weiter machen.
7500 data  32, 96,166:rem " ..55  20 60
a6  jsr a660      CLR-Befehl
7510 data  32, 51,165:rem " ..58  20 33
a5  jsr a533      BASIC-Zeilen binden
7520 data  76,133,227:rem " ..5b  4c 85
e3  jmp e385      in READY-Modus springen
7530 :
8000 rem " Transport-Routinen: 1 BYTE Se
nden:     $..5e
8010 :
8100 data 141,  1,221:rem " ..5e  8d 01
dd  sta dd01      AKKU in USER-Port
8110 data 162,147:     rem " ..61  a2 93
    ldx#93        VALID-Signal senden
8115 data 142,  0,221:rem " ..63  8e 00
dd  stx dd00
8120 data 173, 13,221:rem " ..66  ad 0d
dd  lda dd0d      QUITTUNG laden
8125 data 208, 11:     rem " ..69  d0 0b
    bne ..76      QUITTUNG empfangen
8130 data 202:         rem " ..6b  ca
    dex           abzaehlen der Zeit
```

```
8135 data 208,248:     rem " ..6c  d0 f8
    bne ..66      Ueberschreitung?
8140 data  32,143, -1:rem " ..6e  20 8f
..  jsr ..8f      USER-Port als Eingang
8150 data 162,  5:     rem " ..71  a2 05
    ldx#05        Fehler-Nummer Bet.System
8155 data  76, 58,164:rem " ..73  4c 3a
a4  jmp a43a      Fehler-Routine Rechner
8160 data 162,151:     rem " ..76  a2 97
    ldx#97        VALID-Flag aus
8165 data 142,  0,221:rem " ..78  8e 00
dd  stx dd00      VALID zuruecknehmen
8170 data  96:         rem " ..7b  60
    rts           RETURN
8180 :
8200 rem " Transport-Routinen: 1 BYTE Em
pfangen: $..7c
8202 :
8204 data 173, 13,221:rem " ..7c  ad 0d
dd  lda dd0d      VALID holen
8210 data 240,251:     rem " ..7f  f0 fb
    beq ..7c       ''  warten
8220 data 173,  1,221:rem " ..81  ad 01
dd  lda dd01      BYTE  holen
8230 data 162,147:     rem " ..84  a2 93
    ldx#93        QUITTUNG-Signal senden
8235 data 142,  0,221:rem " ..86  8e 00
dd  stx dd00
8240 data 162,151:     rem " ..89  a2 97
    ldx#97        QUITTUNG-Signal loeschen
8245 data 142,  0,221:rem " ..8b  8e 00
dd  stx dd00
8250 data  96:         rem " ..8e  60
    rts           RETURN
8260 :
9000 rem " Richtungs-Register auf Eingan
g:  $..8f
9010 rem "                         Ausgan
g:  $..92
9020 :
9100 data 162,  0:     rem " ..8f  a2 00
    ldx#00        USER-Port als Eingang
9110 data  44,162,255:rem " ..92  a2 ff
    ldx#ff        USER-Port als Ausgang
9120 data 160,151:     rem " ..94  a0 97
    ldy#97
9130 data 140,  0,221:rem " ..96  8c 00
dd  sty dd00      VALID-Flag zuruecksetzen
9140 data 142,  3,221:rem " ..99  8e 03
dd  stx dd03      USER-Richtung setzen
9150 data  96:         rem " ..9c  60
    rts           RETURN
9160 :
9200 rem " Initialisierung des Ports
    $..9d
9210 :
9220 data 173, 13,221:rem " ..9d  ad 0d
dd  lda dd0d      VALID laden
9230 data 208,251:     rem " ..a0  d0 fb
    bne ..9d      Flag gesetzt:noch einmal
9240 data  76,143, -1:rem " ..a2  4c 8f
..  jmp ..8f      USER-Port als Eingang

ready.
```

# Drucker/ Floppy ein- oder ausgeschaltet?

## Diese Prüfroutine verhindert wirkungsvoll die Fehlermeldung: DEVICE NOT PRESENT ERROR und hilft, ein Programm absturzsicher zu machen.

Bei der Prüfroutine handelt es sich um ein Unterprogramm, das mit GOSUB 10080 aufgerufen wird. Vor dem Aufruf der Routine muß festgelegt werden, welches Gerät überprüft werden soll (Gerätenummer DN%: 4 für Drucker, 8 für Floppy). Die Auswertung der Variablen DR% (0 = aus, 1 = ein) ermöglicht eine Aussage über den Einschaltzustand.

**Prüfroutine**

Zeile 10080:
Um die Maschinenroutine nicht bei jedem Aufruf des Unterprogramms einzulesen, wird sie nur beim ersten Aufruf eingelesen.

Zeile 10090 und 10100:
In die vom Computer nicht benutzte Speicherstelle 2 der Zeropage wird die Device-Nummer DN% (4 für Drucker, 8 für Floppy) gePOKEt, das Maschinenprogramm aufgerufen und anschließend der Inhalt der Speicherstelle 2 überprüft. War das Carry-Bit gesetzt, ist das entsprechende Gerät ausgeschaltet. Die Variable DR% wird entsprechend dem Einschaltzustand 1 (= ein) oder 0 (= aus) gesetzt.

Zeile 10140 bis 10170:
Basic-Lader der Maschinenroutine

Nach dem Löschen des Bildschirms wird der Variablen DN% die entsprechende Geräte-Nummer (4 für Drucker, 8 für Floppy) zugewiesen und die Prüfroutine aufgerufen. Die Abfrage der Variablen DR% (Device ready) ermöglicht es, eine Aussage über den Einschaltzustand (0 = aus, 1 = ein) des entsprechenden Gerätes zu machen.

Wird die Prüfroutine beim Diskettenlaufwerk angewendet (DN%=8), so muß nach dem Aufruf der Routine der Fehlerkanal der Floppy gelöscht werden, da durch die Prüfung ein Syntax Error (31) vom DOS gemeldet wird (Ansprung des NMI-Vektors in Zeile 160).

(Werner Pfeil/gk)

```
10 PRINT"♥       WERNER PFEIL"
20 PRINT"        AM MALZBUECHEL 4"
30 PRINT"  5000 KOELN 1"
40 PRINT"        0221/24 25 53"
50 PRINT:PRINT"            64'ER MAGAZIN
8/84"
51 PRINT"SERIELLER BUS - TEST":PRINT
52 PRINT"UEBERPRUEFUNG OB DRUCKER ODER
53 PRINT"FLOPPY EIN/AUSGESCHALTET SIND"
60 PRINT:PRINT:PRINT"     TASTE"
70 POKE198,0:WAIT198,1
100 REM * DEM-PROGRAMM *
110 :
120 PRINT CHR$(147)
130 DN%=4:GOSUB 10080:REM DRUCKER
140 IF DR%=1THENPRINT"DRUCKER EINGESCHAL
TET"
150 IF DR%=0THENPRINT"DRUCKER AUSGESCHAL
TET"
160 DN%=8:GOSUB 10080:IF DR%=1 THEN OPEN
1,8,15,"UI:":CLOSE 1:REM FLOPPY
170 IF DR%=1THENPRINT"FLOPPY  EINGESCHAL
TET"
180 IF DR%=0THENPRINT"FLOPPY  AUSGESCHAL
TET"
190 END
200 :
210 :
220 :
10000 REM * PRUEFROUTINE DEVICE EIN? *
10070 :
10080 IF FT%=0 THEN FT%=1:GOSUB 10140:RE
M ROUTINE NUR BEIM 1. MAL EINLESEN
10090 POKE 2,DN%:SYS49152:IF PEEK(2)=0 T
HENDR%=1:RETURN:REM DEVICE 'EIN'
10100 DR%=0:RETURN:
      REM DEVICE 'AUS'
10110 :
10120 REM * MASCHINENROUTINE EINLESEN *
10130 :
10140 FOR AD=49152TO49185:READ A:POKE AD
,A:NEXT AD
10150 DATA169,1,160,192,162,33,32,189,25
5,169,1,160,15,166,2,32,186,255,32
10160 DATA192,255,176,2,169,0,133,2,169,
1,32,195,255,96,32
10170 RETURN
READY.
```

**Ein Demo-Programm. Die Prüfroutine befindet sich ab Zeile 10000**

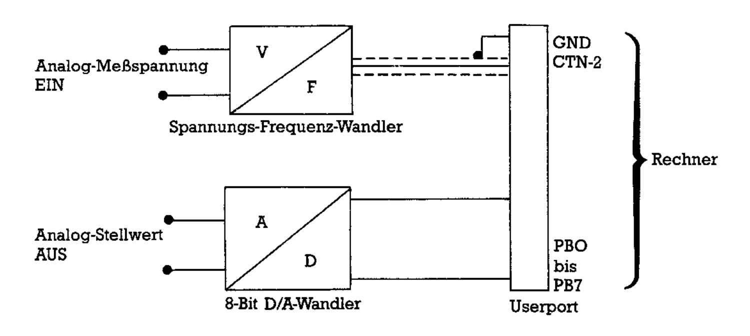
```
                0010     ; ****************************
                0020     ; *   ASSEMBLER-LISTING DER   *
                0030     ; * PRUEFROUTINE DEVICE EIN ? *
                0040     ; *  GESCHRIEBEN FUER  C- 64  *
                0050     ; * (C) 1984 BY  WERNER PFEIL *
                0060     ; ****************************
                0070     ;
                0100     .BA $C000
C000- A9 01     0110     LDA #$01        ; LAENGE
C002- A0 C0     0120     LDY #$C0        ; ADRESSE HIGH
C004- A2 21     0130     LDX #$21        ; ADRESSE LOW
C006- 20 BD FF  0140     JSR $FFBD       ; PARAMETER FUER FILENAMEN
C009- A9 01     0150     LDA #$01        ; FILE-NR.
C00B- A0 0F     0160     LDY #$0F        ; SEKUNDAERADRESSE
C00D- A6 02     0170     LDX $02         ; DEVICE-NR.
C00F- 20 BA FF  0180     JSR $FFBA       ; FILEPARAMETER SETZEN
C012- 20 C0 FF  0190     JSR $FFC0       ; OPEN
C015- B0 02     0200     BCS $C019       ; WENN AUSGESCHALTET, DANN
C017- A9 00     0210     LDA #$00
C019- 85 02     0220     STA $02         ; ERGEBNIS DER PRUEFUNG
C01B- A9 01     0230     LDA #$01        ; FILE-NR.
C01D- 20 C3 FF  0240     JSR $FFC3       ; CLOSE
C020- 60        0250     RTS
C021- 20        0260     .BY $20
                0270     .EN
```

**Der Inhalt der DATA-Zeilen mit zugehörigem Assembler-Listing**

# Analoger Meßwert rein — analoger Stellwert raus

**In diesem Beitrag wird eine einfache Anordnung für den Commodore 64 beschrieben, bei der ohne Umschalten ein Analogsignal eingegeben und ein Analogstellwert ausgegeben werden kann.**

Für einen regeltechnischen Versuch wurde eine Anordnung gebaut, bei der über eine dauernd mit dem Computer verbundene Meßleitung ein analoges Meßsignal aufgenommen, im Computer verarbeitet und über die 8-Bit-Datenleitung des Userports und einen 8-Bit-D-A-Wandler wieder ein analoges Stellsignal als Antwort ausgegeben werden kann (siehe Bild)




### Ein- und Ausgangsschaltung

Diese Anordnung dürfte auch für viele Bastelzwecke sehr interessant sein. Für den Analogeingang wurde mit dem IC LM311 ein Spannungs-Frequenzwandler gebaut. Die Frequenz des erzeugten Rechtecksignals wurde über eine einadrige Schirmleitung mit Ader auf PIN6 = CNT-2 des Userports und Schirm auf PIN1 = GND zur Frequenzmessung eingegeben. Diese Anwendung ist prinzipiell in [1] beschrieben, allerdings mit dem kleinen Schönheitsfehler, daß die Sache so nicht funktioniert. Trotzdem ist das erwähnte Buch sehr empfehlenswert wegen seiner Vielfalt und Verständlichkeit. Im folgenden wird zunächst ein funktionsfähiges Programm gezeigt und anschließend werden wesentliche Verbesserungen unter Verwendung einer Assembler-Routine geschildert. Ein Studium der Literatur [1 bis 4] zeigt, daß für eine einfache Frequenzmessung bis hinauf zu etwas mehr als 20 kHz der im CIA 6526 (Complex Interface-Adapter-IC) eingebaute Zähler und die Systemuhr TI als Zeitbasis in Frage kommen. Wer einen Rechteckgenerator zur Hand hat, kann dies auch gleich (TTL-Pegel 5 V einstellen) mal ausprobieren! Geben sie das folgende Basic-Programm ein und schließen Sie den Generator — wie beschrieben — an CNT-2 an.

```
90 REM****BASIC-PROGRAMM FREQUENZMESSUNG************
100 POKE56590,240
110 POKE56580,255:POKE56581,255
120 T=TI
130 IFT=TITHEN130
140 POKE56590,241
150 IFTI<>T+60THEN150
160 A=PEEK(56581):B=PEEK(56580)
170 PRINT65535-(256*A+B)
180 GOTO100:REM !!DAUERSCHLEIFE!
```

**Listing »Frequenzmessung«**

Es wurden folgende Adressen verwendet:
Dez. 56590 Hex DD0E Kontrollregister für Zähler 1
Dez. 56580 Hex DD04 Zähler 1 Low-Byte T1L
Dez. 56581 Hex DD05 Zähler 1 High-Byte T1H

In Zeile 100 wird das Kontrollregister für den Zähler 1 so programmiert, daß dieser in der gewünschten Art arbeitet und zunächst im STOP-Zustand ist. In Zeile 110 wird der Zähler mit HB = 255 und LB = 255 geladen; er hat die Eigenschaft, diese zwei Byte nach dem Start herunterzuzählen (Dez. 65535 bis 0). Der Zähler kann nun aber nicht, wie in [1] erwähnt, mit dem Setzen des High-Byte von T1 (T1H) gestartet werden, sondern durch den Befehl Poke 56590, 241, bei dem Bit 0 im Kontrollregister auf 1 = START gesetzt wird. Die TI-Uhr wird bei diesem Programm »im Vorübergehen« gelesen, die beiden IF-Abfragen geben den Start- und den Auslesebefehl für den Zähler.

Im Kontrollregister haben nur Bit 0 und 5 wichtige Funktionen:
11110001 = Dez. 241 → Zähler Start
11110000 = Dez. 240 → Zähler Stop

Die anderen Bits sind als »Dauermaske« aufzufassen.

Das Basicprogramm zeigt jedoch leider bei der praktischen Anwendung Anzeigeschwankungen von etwa 2 Prozent und manchmal weit darüber. Schuld daran ist sicher die Programmsprache Basic, welche zum Beispiel aufgrund von zu langsamen Bearbeitungszeiten des Basic-Interpreters in den IF-Abfragen ziemliche Fehler in der Zeitbasis zu verursachen scheint. Gelegentlich kommt die ganze Messung aber auch noch stärker »aus dem Tritt«. Geht man dazu über, die Uhr TI nicht »im Vorübergehen« abzulesen, dann kann man schon eine etwas bessere Genauigkeit erreichen.

```
90 REM****BASIC-PROGRAMM2 FREQUENZMESSUNG************
100 POKE56590,240
110 POKE56580,255:POKE56581,255
120 POKE160,0:POKE161,0
130 POKE162,0:POKE56590,241
150 IFTI<>T+60THEN150
160 A=PEEK(56581):B=PEEK(56580)
170 PRINT65535-(256*A+B)
180 GOTO100:REM !!DAUERSCHLEIFE!
```

**Listing »Frequenzmessung« 2**

In diesem Programm wird in Zeile 120/130 durch EinPOKEn von 0 in die drei TI-Register 160, 161, 162 die Uhr TI jedesmal neu gestartet, wobei Register 162 des »schnellsten Rädchens« zuletzt genullt wird. Hierdurch entfällt die erste IF-Schleife des Starts. Benützt wird im Prinzip nur Regsiter 162, in welchem die Uhr von 0 bis 255 in Schritten von 1/60 s (Jiffies) läuft. Die beiden anderen Register werden nur zur Sicherheit genullt.

Ausgehend von diesem Programm wurde das folgende Assembler-Programm geschrieben, das im Prinzip den gleichen Ablauf hat, bei dem aber die Startbefehle für Zähler und Uhr TI sowie der Stop-Befehl für den Zähler und vor allem die Zeitabfrage und das Ablesen »blitzschnell« ablaufen:

```
DD0E                100 CRA    =    $DD0E
DD04                110 T1L    =    $DD04
DD05                120 T1H    =    $DD05
00A2                130 TIM    =    $A2
00A0                131 TY1    =    $A0
00A1                132 TY2    =    $A1
C000                140        *=    $C000
C000   A9 F0        150        LDA   #$F0
C002   8D 0E DD     160        STA   CRA
C005   A9 00        220        LDA   #$00
C007   85 A0        221        STA   *TY1
C009   85 A1        222        STA   *TY2
C00B   85 A2        230        STA   *TIM
C00D   A9 F1        240        LDA   #$F1
C00F   8D 0E DD     250        STA   CRA
C012   A5 A2        260 MA     LDA   *TIM
C014   C9 3C        270        CMP   #$3C
C016   D0 FA        280        BNE   MA
C018   AE 05 DD     290        LDX   T1H
C01B   AC 04 DD     300        LDY   T1L
C01E   60           320        RTS
                    330        .EN
```

**Listing »Frequenzmessung Maschinensprache«**

```
100 FOR I = 49152 TO 49182
110 READ X:POKEI,X:S=S+X:NEXT
120 DATA169,240,141, 14,221,169,  0,133,160,133,161,133
130 DATA162,169,241,141, 14,221,165,162,201, 60,208,250
140 DATA174,  5,221,172,  4,221, 96
150 IFS<> 4561 THEN PRINT"FEHLER IN DATAS":END
160 PRINT"OK"
```

**Listing »Basiclader«**

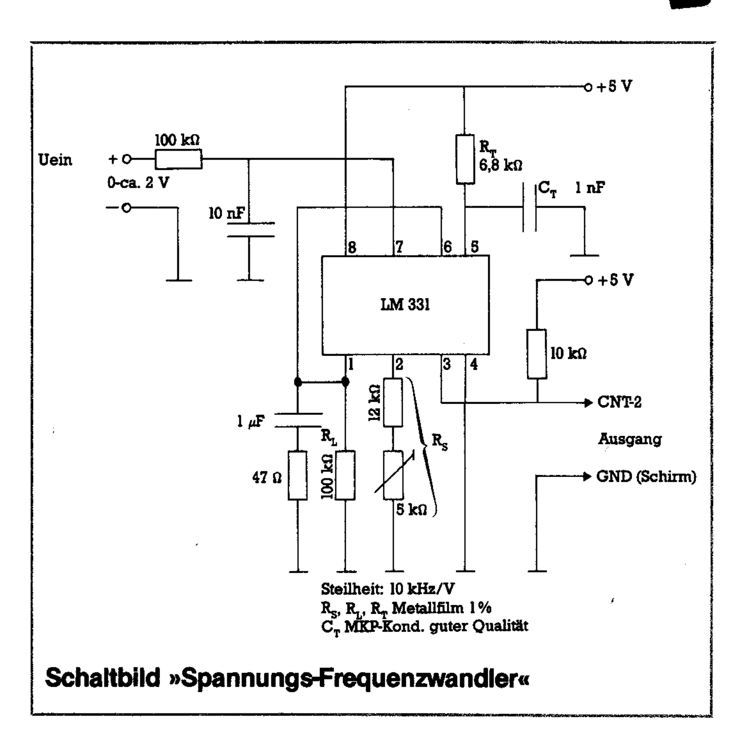
Nach einmaligem RUN-Lauf kann dieses Programm immer mit SYS 49152 aufgerufen werden, der Basiclader kann dann gelöscht werden (vorher natürlich speichern!). Das Programm arbeitet einen Durchlauf des Meßzyklus ab und kehrt dann nach Basic zurück, zuvor hat es jedoch HB und LB des Zählers, T1H und T1L, nach Adresse Dez 781 und 782 abgelegt. Aus diesem »Briefkasten« können die Werte mit PEEK geholt werden. Als Beispiel ein Miniprogramm zur dauernden Frequenzanzeige in Hz:

```
100 SYS49152
110 A=PEEK(781):B=PEEK(782)
120 PRINT65535-(256*A+B)
130 GOTO100:REM !!DAUERSCHLEIFE!
```

**Listing »Frequenzanzeige«**

Mit diesem Programm wurde eine Genauigkeit von deutlich besser als 1 Prozent erhalten. Versuche zeigten, daß selbst im praktischen Betrieb mit einem Spannungs-Frequenzwandler die Meßdauer ohne Verschlechterung des Ergebnisses auf 30 x 1/60 s + 0,5 s verkürzt werden kann. Dies kann man im Basiclader dadurch bewerkstelligen, daß in Zeile 130 die Zahl 60 in 30 geändert wird.

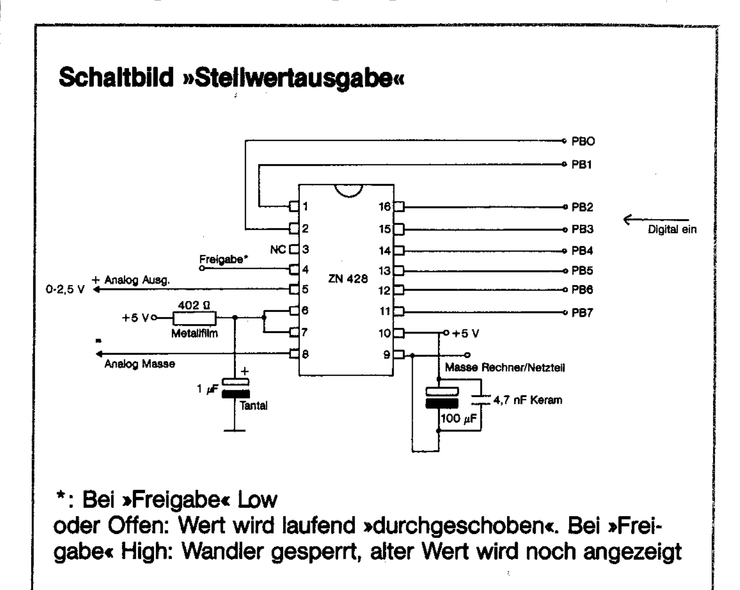
Bei der praktischen Anwendung wurde entsprechend [1] beziehungsweise nach Angaben des Herstellers [5] mit dem IC LM 331 folgender Spannungs-Frequenzwandler gebaut:




**Schaltbild »Spannungs-Frequenzwandler«**

Die Schaltung funktioniert sehr gut und recht genau! Für die »Stellenwertausgabe« wurde mit dem 8-Bit-D-A-Wandler ZN428 folgende Schaltung aufgebaut:

**Schaltbild »Stellwertausgabe«**




*: Bei »Freigabe« Low oder Offen: Wert wird laufend »durchgeschoben«. Bei »Freigabe« High: Wandler gesperrt, alter Wert wird noch angezeigt

Die Ausgabe über den Userport erfolgt mit Hilfe folgender Befehle:
Poke 56579, 255 (Datenleitungen PBO bis PB7 auf Ausgabe)
Poke 56577, x (x = Dez.-Wert der Ausgabe)
Der Wandler erzeugt dann eine zu x proportionale Spannung.

Dem Nachteil einer V/f-Wandler-Zeit von ≈ 0,5 s steht der Vorteil gegenüber, daß keine Ein-Ausgabeleitungs-Umschaltung erforderlich ist, die Signale stehen stetig am Port zur Verfügung. Eine mehrkanalige Ausführung des Eingangs durch Multiplexen ist denkbar. Beim Ausgang ist dies sicher etwas schwieriger.

(Dr. Ernst Breitz/rg)

Literatur
[1] Anderson A., Kullbjer A.: »Messen, steuern, regeln mit dem VC 20 und Commdore 64«, Haller-Verlag, Saarbrücken

[2] Angerhausen, Brückmann, Englisch, Gerits, »64 intern«, Data-Becker, Düsseldorf

[3] Floegel E., »Hardware-Erweiterungen für Commodore 64«, Hofacker-Verlag, Holzkirchen

[4] Brückmann, »Der Commodore 64 und der Rest der Welt«; Data-Becker, Düsseldorf

[5] National-Semiconductor, Daten-Blatt zu LM 131/231/331

# »Kudiplo« erfüllt Schülerträume

## Eine komplette Kurvendiskussion ist für Computerbesitzer eine ganz einfache Angelegenheit.

Wer hätte sich da nicht gewünscht, solche lästige Arbeit nähme ihm ein fleißiger Computer ab. Diesen Wunsch erfüllt »Kudiplo« jedenfalls für den, der einen VC 20 mit wenigstens 3 KByte Erweiterung besitzt. Für den C 64 ist »Kudiplo« leicht einzurichten. Nur die Bildschirmbefehle sind anzupassen.

Benötigt wird außerdem der VC 1520, der Printer-Plotter von Commodore, der die grafische Darstellung mit einer Präzision möglich macht, die keine Wünsche offen läßt. Mit »Kudiplo« kann dann jede beliebige mathematische Funktion im frei wählbaren Bereich und Maßstab dargestellt werden. Außerdem wird auf Wunsch eine Kurvendiskussion ausgedruckt mit Nullstellen, Sprungstellen und Extrema, deren Koordinaten bis zur vierten Dezimalstelle genau berechnet werden.

Hier die besonderen Fähigkeiten des Programmes:

☐ Auf beiden Koordinatenachsen kann der dargestellte Bereich frei gewählt werden durch Bestimmung des jeweils untersten und obersten Wertes.

☐ Die zweckmäßigste Positionierung der Koordinatenachsen wird automatisch ermittelt (Normalfall: Kreuzung im Nullpunkt).

☐ Entsprechend dem gewählten Bereich wird die Skalierung des Achsenkreuzes berechnet und in die Grafik geschrieben.

☐ Die Maßeinheit beider Achsen beträgt dabei immer 10 mm, so daß Zwischenwerte leicht aus der Grafik ausgemessen und berechnet werden können.

☐ Der frei wählbare Darstellungsbereich macht es möglich, durch Wahl eines Teilbereiches Ausschnittvergrößerungen der Funktion in jedem beliebigen Maßstab darzustellen (!).

☐ In ein Achsenkreuz kann nach der ersten auch noch eine zweite und beliebig oft noch eine weitere Funktion (etwa zum Vergleich mit der ersten) geplottet werden — solange nicht mit »Paper feed« der Papiertransport betätigt wurde.

☐ Jede Funktion wird in anderer Farbe gezeichnet und die zugehörige Gleichung mit gleicher Farbe in die Grafik eingetragen.

☐ Auf Wunsch wird von der zuletzt dargestellten Funktion die Kurvendiskussion erstellt.

☐ Dabei werden ermittelt: Extrema, Maxima, Minima und Sattelpunkte sowie die Nullstellen unter Angabe der Steigung und die Sprungstellen mit ihren Koordinaten.

☐ Verarbeitet werden alle dem VC 20- (beziehungsweise C 64) geläufigen mathematischen Funktionen. Bei ihrer Eingabe ist nur auf die Einhaltung der üblichen Syntax zu achten.

☐ Ein besonderer Vorzug von »Kudiplo« ist schließlich, daß das Programm unzulässige Werte erkennt und unberücksichtigt läßt. SQR(X) führt darum bei Werten von X < 0 ebensowenig zum Abbruch des Programms wie 1/X bei X = 0.

Die Gebrauchsanleitung für den Benutzer wird über den Bildschirm angegeben. Unlogische Eingaben werden zurückgewiesen.

Das Programm erfragt zuerst den Bereich, in welchem die Funktion dargestellt wird (Zeilen 470 bis 540). Die entsprechenden Werte werden in den Variablen XU, XO, YU und YO gespeichert. Aus diesen Werten wird die Lage des Nullpunktes auf beiden Achsen (XN und YN) und die Maßeinheit für die Skalierung der Achsen (XE und YE) berechnet. Weiter wird anschließend die darzustellende Funktion in den Zeilen 585 bis 620 erfragt.

Die folgende Routine dient dazu, die Funktion in das Programm zu übernehmen, indem sie zusammen mit ihrer Ableitung in die Programmzeilen 65 bis 80 und 205 geschrieben wird. Die Routine bedient sich dabei des bekannten Verfahrens der Eingabe über Bildschirmspeicher und Tastaturpuffer:

Die zu verändernden Zeilen werden auf den Bildschirm geschrieben, dann wird der Tastaturpuffer mit der erforderlichen Zahl von »RETURNs« gefüllt. Bei dem dann folgenden Programmabbruch (Zeile 660) arbeitet der Computer die Befehle im Tastaturpuffer ab. Damit bei der hiermit bewirkten Änderung des Programmes die zuvor gewählten Parameter nicht verloren gehen, werden auch diese (mit Eingabe in die Zeilen 45 und 50) gerettet. Damit diese Routine funktioniert, dürfen die Zeilennummern 45, 50, 65, 70 und 205 nicht verändert werden!!

Nachdem sich das Programm durch »GOTO 45« selbst wieder gestartet hat, werden zunächst die Parameter aus den Zeilen 45 bis 55 und aus den Zeilen 65 bis 80 die definierten Funktionen eingelesen.

Nun endlich kann der Plotter in Funktion treten. Er zeichnet die X-Achse (Zeilen 100 bis 140) und die Y-Achse (Zeilen 145 bis 165) in der Größe von 400 x 400 Druckersteps (= 8 x 8 cm).

Es folgt in den Zeilen 195 bis 225 die Berechnung der Funktionswerte, beginnend mit dem niedrigsten Wert von X und mit der Schrittweite J von normalerweise 1/400 des Gesamtbereichs. Wer die Ausgabe (auf Kosten der Auflösung) beschleunigen möchte, erreicht dies durch Zuweisung eines höheren Wertes für J.

Die Zeilen 185 bis 195 bedürfen einer etwas ausführlichen Erklärung: Normalerweise würde das Auftreten unzulässiger Werte in der Funktion von X zum Abbruch des Programmes führen, so zum Beispiel wenn der Logarithmus von Werten X<0 gebildet werden oder durch X=0 dividiert werden soll. Die Verarbeitung solcher Funktionen erfordert einen Eingriff ins Betriebssystem: Der Zeiger für die Fehlerbehandlungs-Routine (Speicherplätze $0300 und $0301) wird so (mit Zeile 190) verbogen, daß er jetzt auf einen in den Kassettenpuffer gePOKEten (Zeile 185) Sprungbefehl zeigt.

Infolge dieses Eingriffs gibt nun das Betriebssystem bei Auftreten eines Fehlers nicht mehr Alarm, sondern beginnt mit der Abarbeitung der nächsten (!) Zeile. Restliche Statements in der fehlerträchtigen Zeile werden also nicht mehr beachtet.

Mit diesem Trick wird bewirkt, daß der Sprungbefehl am Ende der Zeile 205 normalerweise ausgeführt wird, bei Auftreten eines Fehlers aber die Zeilen 210 bis 215 abgearbeitet werden. Hier wird je nach Art des Fehlers die Null-Fehler-Flag FE gesetzt oder der unzulässige Wert von X in der Variablen XW festgehalten. Diese Werte sind im weiteren Programmablauf behilflich, unzulässige Werte von X zu vermeiden.

Nach Berechnung aller Werte von X und Ausdruck der entsprechenden Punkte der Funktion (X;Y) in Zeile 220 wird mit den POKE-Befehlen der Zeile 225 der Zeiger für die Fehlerausgabe wieder in den Normalzustand gebracht. Die eingegebene Funktion ist damit fertig geplottet. Das Papier wird vorgeschoben, damit das Ergebnis besichtigt werden kann (Zeile 235). Auf dem Bildschirm erscheint zugleich das Menü (Zeilen 665 bis 720). Abhängig von der Eingabe verzweigt das Programm nun entweder

1) zur Eingabe einer Funktion, die in die gleichen Koordinaten geplotten werden soll, oder

2) zur Eingabe der Parameter für ein neues Koordinatenkreuz oder

3) zur Ausgabe der Kurvendiskussion, die jeweils nur von der letzten dargestellten Funktion erstellt werden kann.

Im Rahmen der Routine für die Berechnung der Kurvendiskussion werden die Fehler-Flags XW und FE dazu verwendet, unzulässige Werte auszulassen.

Nun werden zuerst die Null- beziehungsweise Sprungstellen (Zeilen 285 bis 360) und dann die Extrema (Zeilen 365 bis 390) ermittelt. Doch nun viel Spaß beim Arbeiten mit »Kudiplo«.

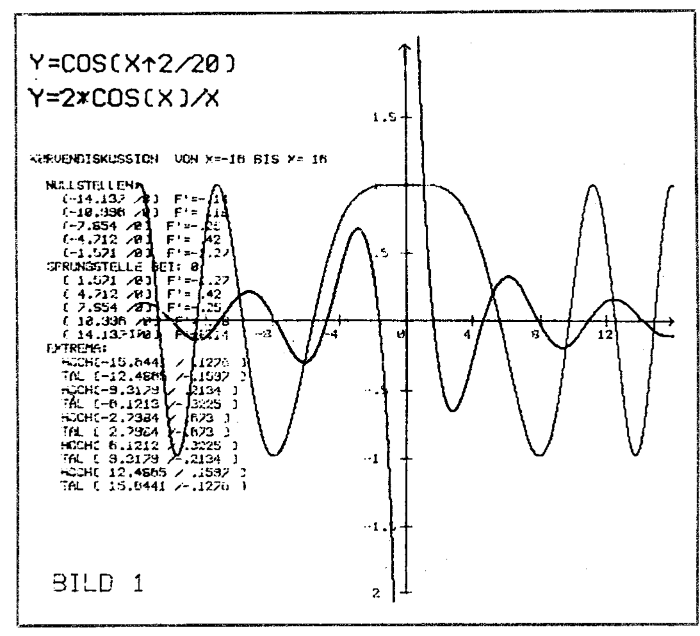
(Jürgen Curdt/ev)

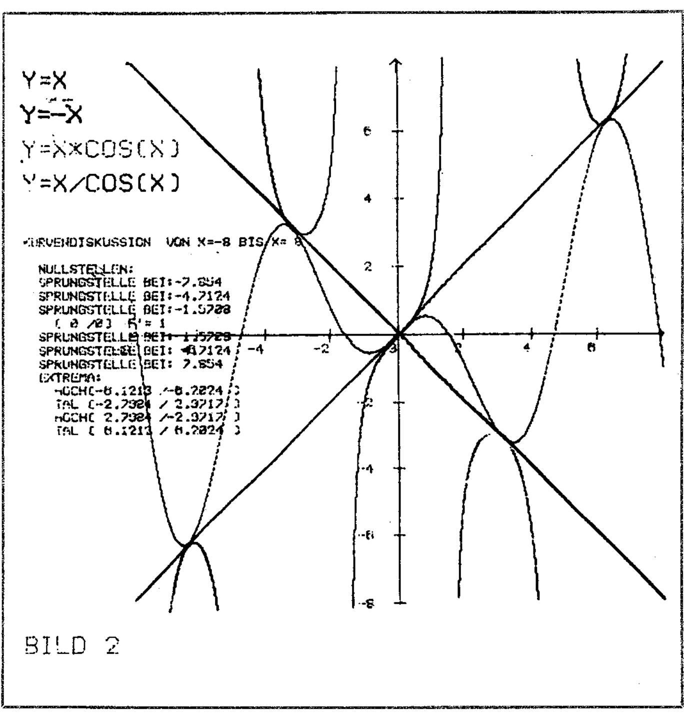
```
5 REM***********KUDIPLO****************
10 REM FUNKTIONEN DISKUTIEREN UND
12 REM PLOTTEN MIT VC-20 +3K
15 REM UND PRINTER-PLOTTER VC-1520
20 REM JUERGEN CURDT
25 REM KESSEMEIERWEG 5
30 REM 493 DETMOLD
35 REM*********************************
40 POKE36879,25:GOTO45
45 XU=-4:XO= 4:YU=-4:YO= 4:OY= 0
50 XN= 280:YN= 0
55 XE=(XO-XU)/16:YE=(YO-YU)/16
60 REM FUNKTION BIS 80
65 DEFFNF(X)=HIER WIRD DIE FUNKTION EING
EFUEGT
70 F$="HIER WIRD DIE FUNKTION EINGEFUEGT
"
75 DEFFNF1(X)=(FNF(X+1E-4)-FNF(X-1E-4))/
2E-4
80 DEFFNF2(X)=(FNF1(X+1E-4)-FNF1(X-1E-4)
)/2E-4
85 OPEN1,6,1:OPEN2,6,2:OPEN3,6,3:OPEN10,
6
90 PRINT"■";:IFP=0GOTO550
95 REM  KREUZ ZEICHNEN
100 PRINT#3,0:PRINT#2,1
105 IFSVTHEN170
110 PRINT#10:PRINT#1,"M";0;-440:PRINT#1,
"M";0;-200:PRINT#10
115 PRINT#1,"M";80;YN:FORI=0TO14STEP2
120 PRINT#1,"I":PRINT#1,"R";0;4:PRINT#1,
"J";0;-4
125 PRINT#1,"R";-12;-14
130 PRINT#10,INT((XE*I+XU)*100+.5)/100;
135 PRINT#1,"M";80+I*25;YN:PRINT#1,"D";8
0+(I+2)*25;YN
140 NEXT:PRINT#1,"M";75+I*25;YN-4:PRINT#
10,">";
145 PRINT#1,"M";XN;-200
150 FORI=0TO14STEP2:PRINT#1,"I":PRINT#1,
"R";4;0:PRINT#1,"J";-4;0:PRINT#1,"R";-30
;-4
155 J=YU+INT(YE*100*I+.5)/100:IFJ<>0THEN
PRINT#10,J;
160 PRINT#1,"M";XN;I*25-200:PRINT#1,"D";
XN;(I+2)*25-200
165 NEXT:PRINT#1,"M";XN-5;I*25-209:PRINT
#3,1:PRINT#10,"^";
170 REM KURVE PLOTTEN
175 PRINT"■■■■■  ETWAS GEDULD BITTE":PR
INT#2,SV+3
180 E$="M":FE=0:XW=XU
185 POKE832,76:POKE833,59:POKE834,201
190 POKE768,64:POKE769,3
195 J=(XO-XU)/400:FORI=0TO400
200 X=J*I+XU
205 Y=HIER WIRD DIE FUNKTION EINGEFUEGT:
GOTO220
210 E$="M":IFX<0THENXW=X
215 IFX=0THENFE=1:GOTO230
220 Y=Y/YE*25+OY:IFY>210ORY<-210THENE$="
M":GOTO230
225 PRINT#1,E$;I+80;Y:E$="D"
230 NEXT:POKE768,58:POKE769,196
235 PRINT#1,"M";0;180-SV*25:PRINT#3,1:PR
INT#10,"Y="F$;
240 PRINT#1,"M";0;-250
245 GOTO665
250 PRINT#1,"M";0;160-SV*25
255 PRINT"■DIE KURVENWERTE WERDEN
BERECHNET
260 PRINT#10:PRINT#3,0:IFNOTFEANDXU=XWTH
EN280
265 IFFEANDXW=XUTHENPRINT#10,"UNZULAESSI
GER WERT BEI 0":GOTO275
270 PRINT#10,"UNZULAESSIGE WERTE IM BERE
ICH":PRINT#10,"VON "XU" BIS 0 !!  -   DA
RUM NUR"
275 XU=XW+.1
280 PRINT#10,"KURVENDISKUSSION  VON X="X
U"BIS X="XO:PRINT#10
285 PRINT#10,"  NULLSTELLEN:"
290 A=FNF(XU+.1):S1=XE/40
295 FORX=XUTOXOSTEPS1
300 IFX=0ANDFE=1THEN325
305 F=FNF(X)
310 IFF*A<0THEN335
315 IFABS(F)<1E-6THENC=X:GOTO465
320 A=F
325 NEXTX
330 GOTO370
335 S=X-.1:Z=X:C=X-.05
340 IFFNF(C)*FNF(Z)<0THENS=C:GOTO350
345 Z=C
350 C=(S+Z)/2
355 IFABS(Z-S)<1E-6ANDABS(FNF(C))<1E-3TH
EN465
360 IFABS(Z-S)<1E-6THENPRINT#10,"  SPRUN
GSTELLE BEI:"INT(C*1E4+.5)/1E4:GOTO320
365 GOTO340
370 PRINT#10,"  EXTREMA:":A=FNF1(XU+.1):
FORX=XUTOXOSTEPS1
375 IFX=0ANDFE=1THEN395
380 F=FNF1(X)
385 IFF*A<0THEN405
390 IFABS(F)<1E-6THENC=X:GOTO440
395 A=F:NEXTX
400 PRINT#1,"M";0;-275:CLOSE1:CLOSE2:CLO
SE3:CLOSE10:END
405 S=X-.1:Z=X:C=X-.05
410 IFFNF1(C)*FNF1(Z)<0THENS=C:GOTO420
415 Z=C
420 C=(S+Z)/2
425 IFABS(Z-S)<1E-6ANDABS(FNF1(C))<1E-2T
HEN440
430 IFABS(Z-S)<1E-6THEN395
435 GOTO410
440 IFABS(FNF2(C))<1E-5THENPRINT#10,"
 SATTEL";:GOTO455
445 IFFNF2(C)>0THENPRINT#10,"    TAL  ";:
GOTO455
450 IFFNF2(C)<0THENPRINT#10,"    HOCH";
455 PRINT#10,"("INT(C*1E4+.5)/1E4"/"INT(
FNF(C)*1E4+.5)/1E4"■)":GOTO395
460 REM NULLSTELLEN DRUCKEN
465 PRINT#10,"     ("INT(C*1E3+.5)/1E3;"/
0)  F'=";INT(FNF1(C)*1E2+.5)/1E2
470 GOTO320
475 REM PARAMETER WAEHLEN
480 PRINT" DIE  FUNKTION  WIRD  DARGESTE
LLT IM BEREICH"
485 PRINT"■VON" XU,"BIS"XO:PRINT"AUF DER
```

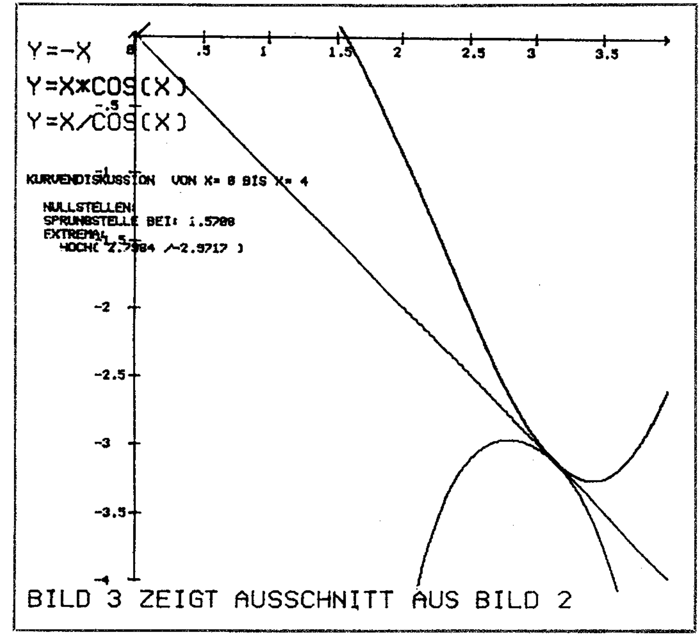
Listing von »Kudiplo«

```
 X-ACHSE"
490 PRINT"■VON" YU,"BIS"YO:PRINT"AUF DER
 Y-ACHSE"
495 PRINT"■■ SOLLEN DIE PARAMETER   VERA
ENDERT WERDEN ?            J/N"
500 GETA$:IFA$="N"THENRETURN
505 IFA$<>"J"THEN500
510 PRINT"■■UNTERSTER WERT FUER X":INPUT
XU
515 PRINT"■■OBERSTER WERT FUER X":INPUTX
O
520 XE=(XO-XU)/16:IFXE<=0THENPRINT"■■  U
NZULAESSIGER WERT  ■":GOTO510
525 IFSGN(XU)=SGN(XO)THENXN=80:GOTO535
530 XN=ABS(XU)*25/XE+80
535 PRINT"■■UNTERSTER WERT FUER Y":INPUT
YU
540 PRINT"■■OBERSTER WERT FUER Y":INPUTY
O
545 YE=(YO-YU)/16:IFYE=<0THENPRINT"■■  U
NZULAESSIGE WERTE  ■":GOTO535
550 REM EINGABE DER       FUNKTION
555 PRINT"■■FUNKTIONEN PLOTTEN UND  DISK
USSION DRUCKEN  ■ MIT DRUCKER VC  1520"
560 FORI=0TO21:PRINT"■";:NEXT:PRINT"■"
565 IFNOTPTHENP=1:GOSUB475:PRINT"■"
570 PRINT"■■■■■■ALS FUNKTION WIRD DAR-
     GESTELLT:":PRINT"■■Y=";F$
575 PRINT"■■■  FUNKTION AENDERN ?
   ■ J/N"
580 IFSGN(YU)<>SGN(YO)THEN595
585 IFYO<=0THENYN=200:OY=ABS(YO*25/YE)+2
00:GOTO600
590 IFYU>=0THENYN=-200:OY=-200-YU*25/YE:
GOTO600
595 YN=ABS(YU)*25/YE-200:OY=YN
600 GETA$:IFA$="N"THEN95
605 IFA$<>"J"THEN600
610 PRINT"■■■■■■GIB DIE FUNKTION EIN !"
615 PRINT"■ENTHAELT DIE FUNKTION EINEN D
IVISOR ALS FUN-TION VON X, DANN DEN"
620 PRINT" DIVISOR INSGESAMT IN     KLAM
MERN SETZEN !
625 INPUT"■■■■Y=";F$
630 PRINT"■■65DEFFNF(X)="F$"
635 PRINT"205Y="F$":GOTO215"
640 PRINT"70F$="CHR$(34)F$CHR$(34)"
645 PRINT"45 XU="XU"■:XO="XO"■:YU="YU"■:
YO="YO"■:OY="OY
650 PRINT"50 XN="XN"■:YN="YN:PRINT"SV="S
V":P="P":GOTO45■■"
655 POKE631,19:FORI=632TO639:POKEI,13:NE
XT:POKE198,8
660 END
665 PRINT"■  NEUE AUFGABE ? ■■■":PRINT"
NEUE FUNKTION         IN DIE GLEICHEN"
670 PRINT"KOORDINATEN            DRUCKEN
          G"
675 PRINT"■■NEUE KOORDINATEN       DRUCKE
N           N"
680 PRINT"■■KURVENDISKUSSION       AUSGEB
EN          K"
685 PRINT"ENDE                    E
690 GETA$:IFA$="N"THENRUN
695 IFA$="K"THEN250
700 IFA$="E"THEN400
705 IFA$<>"G"THEN690
710 SV=SV+1
715 PRINT"■";:GOSUB570
720 GOTO170
READY.
```

**Listing von »Kudiplo« (Schluß)**

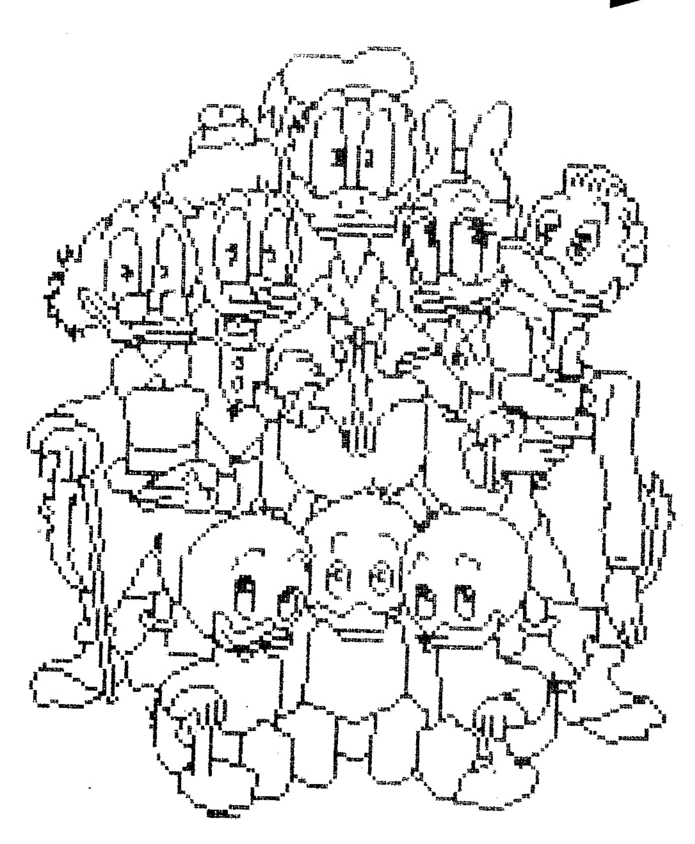


# Hardcopy für den Sieger

**In unserer letzten Ausgabe konnten wir das Görlitz-Interface für Epson-Drucker zum Testsieger erklären. Die Schnittstelle verdankt diesen Sieg hauptsächlich ihrer universellen Anwendbarkeit. Dazu gehört auch das Ausdrucken eines Bildschirminhaltes in Verbindung mit Simons Basic. Daß eine Basicerweiterung aber nicht unbedingt notwendig ist, zeigt diese Hardcopyroutine.**



**Ein Beispiel für den Ausdruck in doppelter Größe**

Mit dem Maschinenprogramm ist es möglich, Grafiken im Maßstab 1:1 (Druckdauer zirka 35 Sekunden, 64000 Dots) oder im Maßstab 2:1 (Druckdauer zirka 2 Minuten, 256000 Dots) ausgeben zu lassen.

Mit dem Basic-Lader kann das Maschinenprogramm in jeden beliebigen Speicherbereich eingelesen werden. Sinnvollerweise sollte das Maschinenprogramm auf die Startadresse #49152 ($C000) gesetzt werden, da hier keine Basic-Bytes verloren gehen. Wird trotzdem eine Endadresse für das Programm festgestellt, die unter #40960 ($A000) liegt, so wird der Beginn des Maschinenprogramms gleichzeitig als Ende des Basic-RAMs ausgewiesen; hierdurch wird verhindert, daß das Maschinenprogramm versehentlich von Basic aus überschrieben wird. Zusätzlich kann wegen der Unterschiedlichkeit der Interfaces gewählt werden, welche Filenummer, welches Gerät und was für eine Sekundäradresse gesetzt werden sollen. Damit die Zeichen ohne Umdefinition vom Computer zum Drucker gesendet werden, ist es je nach Interface notwendig, eine entsprechende Sekundäradresse zu setzen. Entsprechende Hinweise befinden sich sicher bei der Betriebsanleitung des Interfaces. Je nach Bedarf können die Parameter der Initialisierungsroutine geändert werden.

Die mit dem Basic-Lader erzeugte Routine sollte man nach der Erzeugung abspeichern. Für die verschiedenen Speicherkonfigurationen lassen sich natürlich auch verschiedene Hardcopyroutinen erzeugen. Die Startadresse und die Endadresse des erzeugten Maschinenprogramms werden auf dem Monitor angezeigt.

Wird das Maschinenprogramm abgespeichert, so geschieht das unter dem Namen »HC EPSON xxxxx«, wobei xxxxx die Startadresse der Routine wiedergibt.

Hinweise zum Maschinenprogramm:

— Es wird nur eine Datei auf den Drucker eröffnet; diese kann vom Anwender selbst definiert werden.

— Es können die Grafiken aller acht Grafikbildschirme ohne Einschränkungen verarbeitet werden.

— Nach dem Ende des Maschinenprogramms müssen 16 Byte für zwei Umrechnungstabellen freigehalten werden.

— Es werden nur die folgenden Byte in der Zero-Page für die Routine benötigt: $FB-$FF, $02, $A7-$AA.

Zur Implementierung des Programms:

Die Routine kann im Direktmodus, aber auch unter Programmkontrolle aufgerufen werden. Dabei liegt der besondere Vorteil des Programms darin, das sämtliche Parameter durch Basicvariablen ausgedrückt werden können. Das Format zum Ausdruck eines Bildes in Hires-Grafik lautet:

SYS A, B, C.

A — bezeichnet die Startadresse der Routine

B — bezeichnet die Startadresse von einem der acht möglichen Grafikbildschirme

C — gibt den Maßstab der Wiedergabe an. Dabei kann man zwischen vier Ausgabetypen wählen:

0 — gibt das Hiresbild im Maßstab 1:1 aus, zusätzlich wird die Grafik invertiert ausgegeben.

1 — gibt das Hiresbild ohne Invertierung im Maßstab 1:1 aus.

2 — Der Grafikbildschirm wird im Verhältnis 1:2 vergrößert. Zusätzlich wird das Bild invertiert.

3 — Auch hier wird das Bild vergrößert, die Ausgabe des Bildes erfolgt ohne Invertierung.

Wird die Routine als Maschinenprogramm geladen, so muß man nach dem Laden »NEW« eingeben, um sämtliche Zeiger zu korrigieren.

Fehlerbehandlung:

— Werden die Angaben nicht durch Kommata getrennt, erscheint SYNTAX ERROR.

— Wird eine Startadresse für B eingegeben, die nicht mit der Anfangsadresse eines der acht Grafikbildschirme (0, 8192, 16384, 24576, 32768, 40960, 49152 oder 57344) übereinstimmt, so wird je nach Argument »UNDEF'D STATEMENT« oder »ILLEGAL QUANTITY« ausgegeben.

— Bei Angabe eines ungültigen Maßstabs dagegen wird je nach Argument »UNDEF'D FUNCTION« oder »ILLEGAL QUANTITY« ausgegeben.

Anwendungsbeispiel:

Zunächst wird mit dem Basic-Lader die Maschinensprache-routine initialisiert. Soll zum Beispiel die Hardcopyroutine zusammen mit Simons Basic arbeiten, so braucht die vorgesehene Startadresse 49152 nicht geändert zu werden. Die erfragten Parameter Device (Gerätenummer) und Sekundäradresse richten sich nach dem benutzten Interface. Änderungen können bei Bedarf durch Überschreiben der alten Werte vorgenommen werden. Nach korrekter Eingabe wird dann das Maschinenprogramm erzeugt. Nach Abschluß dieser Operation kann dann noch entschieden werden, ob der erzeugte Code abgespeichert werden soll. Wird die Frage positiv beantwortet, so wird das Maschinenprogramm nach Wahl auf Diskette beziehungsweise Kassette abgespeichert. Zum Schluß wird nur noch errechnet, ob das Programm in einem Bereich steht, der von Basic aus erreichbar ist. Ist das der Fall, so wird das Maschinenprogramm vor einem versehentlichen Überschreiben geschützt.

Ist dann die Routine einmal erzeugt, so kann man mit der Ausgabe von Hires-Grafiken beginnen. Erzeugt man zum Beispiel mit Hilfe von Simons Basic eine Hires-Grafik, so wird diese mit SYS xxxxx,57344,1 im Originalmaßstab ausgegeben. Dabei bezeichnet xxxxx die Anfangsadresse der Routine, bei 57344 liegt der Grafikbildschirm von Simons Basic und der letzte Parameter 1 gibt an, daß die Grafik im Maßstab 1:1 ohne Invertierung des Bildes ausgegeben werden soll. Nach ordnungsgemäßer Eingabe des SYS-Befehls beginnt der Drucker das Bild auszugeben.

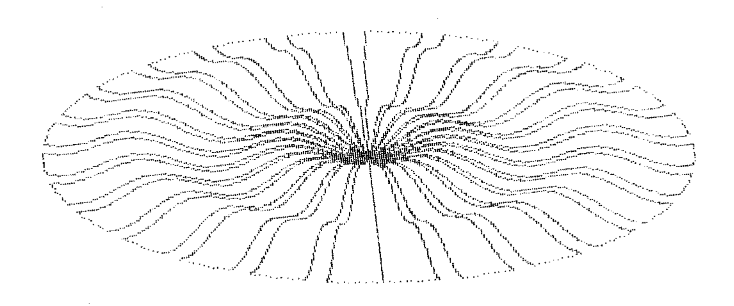
Bei Bedarf kann man dann zu einem späteren Zeitpunkt die abgespeicherte Maschinenroutine nachladen. Liegt sie im Basicbereich (bis #40959,$9FFF), so kann man durch Abtippen und Starten der Zeilen 144 bis 146 des Laders erreichen, daß das Programm vor Überschreibung geschützt wird, ohne gleich den gesamten Lader zu starten. Dazu ist in diesen Zeilen die Variable AN durch die Startadresse der Routine zu ersetzen.

Sonstige Hinweise:

Bevor die Routine beginnt, muß der Drucker bereits eingeschaltet sein.

Werden eventuell andere Codes für die Erzeugung der Hires-Grafik benötigt, so sind die Steuercodes, die die Bit-Image Schreibweise des Druckers steuern, zu ändern. Das hier vorgestellte Maschinenprogramm ist für den FX-80 ausgelegt, ist jedoch auch auf anderen Druckern lauffähig. Da das Assemblerlisting nicht abgedruckt werden kann, können sich Interessierte gerne schriftlich an mich wenden.

(Frank Lonczewski/gk)



**Ein Beispiel für den Ausdruck in einfacher Größe**

```
 100 REM ********************************
**********************
 101 REM *
                              *
 102 REM *  HARDCOPYROUTINE FUER EPSON-D
RUCKER, BASIC-LOADER   *
 103 REM *
                              *
 104 REM *      COPYRIGHT JUNI 1984 BY F
RANK LONCZEWSKI        *
 105 REM * 5810 WITTEN 6,FROEBELSTRASSE
33, TEL.::02302/60933 *
 106 REM * CURSORSTEUERZEICHEN STEHEN IN
 KLAMMERN:"[" UND "]" *
 107 REM ********************************
**********************
 108 PRINT"[CLEAR,RIGHT10,RVSON]EPSON HA
RDCOPYROUTINE[RVOFF,DOWN]"
 109 INPUT"STARTADRESSE[RIGHT7]49152[LEF
T7]";AN:IF AN<0 OR AN>53248 THEN 109
 110 INPUT"FILENUMMER[RIGHT9]127[LEFT5]"
;LF:IF LF<1 OR LF>255 THEN 110
 111 INPUT"DEVICE[RIGHT13]4[LEFT3]";DN:I
F DN<4 OR DN>15 THEN 111
 112 INPUT"SEKUNDAERADRESSE[RIGHT3]4[LEF
T3]";SN:IF SN<0 OR SN>15 THEN 112
 113 I=AN:PRINT"[DOWN]ICH ARBEITE ! BITT
E ETWAS GEDULD."
 114 GOSUB 147
 115 IF RE THEN GOSUB 158:GOTO 117
 116 POKE I,A
 117 PS=PS+A:RE=0:I=I+1:IF I-AN<528 THEN
 114
 118 IF PS<>57824 THEN PRINT"PRUEFSUMMEN
FEHLER !":END
 119 POKE AN+62,DN:REM  DEVICE  SETZEN
 120 POKE AN+64,SN:REM  SEKUN.  SETZEN
 121 POKE AN+60,LF:POKE AN+72,LF:POKE AN
+176,LF:POKE AN+244,LF
    :REM FILENR. SETZEN
 122 PRINT"[CLEAR]EPSON HARDCOPYROUTINE
INITALISIERT"
 123 K=AN+475:B=INT(K/256):K=K-B*256:POK
E AN+47,K:POKE AN+49,B
 124 PRINT"[DOWN]STARTADRESSE[RIGHT4]:";
AN
 125 PRINT"ENDADRESSE[RIGHT6]:";AN+527
 126 PRINT"FILENUMMER[RIGHT6]:";LF
 127 PRINT"DEVICE[RIGHT10]:";DN
 128 PRINT"SEKUNDAERADRESSE:";SN
 129 PRINT"[DOWN]AUFRUF MIT SYS"AN"[LEFT
],GRADR,MAS"
 130 PRINT"[DOWN]GRADR : DEZIMAL GRAFIKB
ILDSCHIRMADRESSE"
 131 PRINT"MAS[RIGHT3]: DEZIMAL MASSSTAB
[DOWN]"
 132 INPUT"CODE ABSPEICHERN[RIGHT4]Y[LEF
```

**Listing der Hardcopy-Routine für Epson-Drucker**

```
T3]";J$:IF J$<>"Y"THEN 144
 133 INPUT"[RVSON]D[RVOFF]ISK ODER [RVSO
N]T[RVOFF]APE[RIGHT3]D[LEFT3]";J$
    :IF J$<>"D"AND J$<>"T"THEN 133
 134 IF J$="D"THEN D=8:GOTO 136
 135 D=1
 136 PS=0:PRINT:FOR I=688 TO 738
 137 GOSUB 147:POKE I,A:PS=PS+A:NEXT
 138 IF PS<>5381 THEN 118
 139 B=AN:H=INT(B/256):L=B-H*256
 140 B=AN+528:HE=INT(B/256):LE=B-HE*256
 141 POKE 707,L:POKE 711,H:POKE 717,LE:P
OKE 719,HE
 142 POKE 693,D:AN$=MID$(STR$(AN),2):FOR
 I=1 TO LEN(AN$)
    :POKE 732+I,ASC(MID$(AN$,I,1))
 143 NEXT I:POKE 157,128:SYS 688:PRINT
 144 IF AN+543>40960 THEN PRINT"[DOWN]MA
SCHINENPROGRAMM WIRD NICHT GESCHUETZT."
    :END
 145 B=AN:H=INT(B/256):L=B-H*256
 146 POKE 56,H:POKE 55,L:CLR:END
 147 READ A$:IF LEN(A$)<>2 THEN PRINT"TI
PPFEHLER IN ZEILE"PEEK(63)+PEEK(64)*256
    :END
 148 A1=ASC(A$):A2=ASC(RIGHT$(A$,1))
 149 IF A1=42 THEN A1=48:RE=1
 150 IF A1<48 OR A1>57 THEN IF A1<65 OR
A1>70 THEN 157
 151 IF A2<48 OR A2>57 THEN IF A2<65 OR
A2>70 THEN 157
 152 IF A1>64 THEN A1=A1-55:GOTO 154
 153 IF A1<58 THEN A1=A1-48
 154 IF A2>64 THEN A2=A2-55:GOTO 156
 155 IF A2<58 THEN A2=A2-48
 156 A=A1*16+A2:RETURN
 157 PRINT"UNGUELTIGER HEXCODE IN ZEILE"
PEEK(63)+PEEK(64)*256:END
 158 B=AN+A2*256:GOSUB 147:I=I+1:B=B+A
 159 HB=INT(B/256):B=B-HB*256:POKE I-1,B
:POKE I,HB:RETURN
 160 DATA 20,FD,AE,20,8A,AD,20,9B
 161 DATA BC,A5,65,F0,03,4C,E3,A8
 162 DATA A5,64,AA,29,10,D0,F6,8E
 163 DATA *0,58,20,FD,AE,20,9E,B7
 164 DATA E0,04,30,03,4C,AE,B3,86
 165 DATA AA,8A,29,80,30,F6,A9,DB
 166 DATA A0,C1,20,1E,AB,A9,00,85
 167 DATA B7,85,A9,A9,7F,A2,04,A0
 168 DATA 04,20,BA,FF,20,C0,FF,A2
 169 DATA 7F,20,C9,FF,A2,06,BD,*1
 170 DATA CD,20,D2,FF,CA,10,F7,A9
 171 DATA 00,85,FC,A9,00,85,FB,18
 172 DATA 69,40,85,FD,A5,FC,69,1F
 173 DATA 85,FE,A5,AA,C9,02,10,05
 174 DATA A9,09,20,D2,FF,A2,04,A5
 175 DATA AA,C9,02,10,0B,BD,*1,C3
 176 DATA 20,D2,FF,CA,10,F7,30,09
 177 DATA BD,*1,C8,20,D2,FF,CA,10
 178 DATA F7,A2,00,86,FF,20,*0,F8
 179 DATA A6,FF,A5,FB,18,69,08,85
 180 DATA FB,A5,FC,69,00,85,FC,E8
 181 DATA E0,28,D0,E7,20,CC,FF,A2
 182 DATA 7F,20,C9,FF,A9,0A,20,D2
 183 DATA FF,A5,AA,C9,02,30,16,A6
 184 DATA A9,D0,12,E6,A9,38,A5,FB
 185 DATA E9,40,85,FB,A5,FC,E9,01
 186 DATA 85,FC,4C,*0,6A,A9,00,85
 187 DATA A9,A5,FC,C5,FE,D0,8B,A5
 188 DATA FB,C5,FD,D0,85,A2,06,BD
 189 DATA *1,D4,20,D2,FF,CA,10,F7
 190 DATA 20,CC,FF,A9,7F,4C,C3,FF
 191 DATA A5,01,48,78,A9,34,85,01
 192 DATA A0,07,B1,FB,99,*2,18,A9
 193 DATA 00,99,*2,10,88,10,F3,68
 194 DATA 85,01,58,20,*1,32,A0,07
 195 DATA A5,AA,C9,02,10,0A,B9,*2
 196 DATA 10,20,*1,B2,88,10,F7,60
 197 DATA B9,*2,10,20,*1,68,88,10
 198 DATA F7,60,A2,07,A0,07,B9,*2
 199 DATA 18,29,80,20,*1,58,18,7D
 200 DATA *2,10,9D,*2,10,88,10,EE
 201 DATA A0,07,B9,*2,18,0A,99,*2
 202 DATA 18,88,10,F6,CA,10,DD,60
 203 DATA 85,02,98,C0,00,F0,05,46
 204 DATA 02,88,10,F7,A8,A5,02,60
 205 DATA 85,A8,98,48,A5,A9,F0,07
 206 DATA A0,04,06,A8,88,D0,FB,A0
 207 DATA 00,84,A7,A5,A8,29,80,20
 208 DATA *1,A6,A5,A8,29,40,20,*1
 209 DATA A5,A5,A8,29,20,4A,20,*1
 210 DATA A5,A5,A8,29,10,4A,4A,20
 211 DATA *1,A5,48,20,*1,B2,68,20
 212 DATA *1,B2,68,A8,60,4A,48,20
 213 DATA *1,AC,68,4A,18,65,A7,85
 214 DATA A7,60,48,A5,AA,29,01,F0
 215 DATA 04,68,4C,D2,FF,68,49,FF
 216 DATA 4C,D2,FF,01,40,00,2A,1B
 217 DATA 02,80,04,2A,1B,00,0D,44
 218 DATA 1B,17,33,1B,0A,0A,0A,0A
 219 DATA 0A,32,1B,0D,48,41,52,44
 220 DATA 43,4F,50,59,20,45,50,53
 221 DATA 4F,4E,0D,0D,28,43,29,20
 222 DATA 20,4A,55,4E,49,20,31,39
 223 DATA 38,34,20,42,59,0D,46,52
 224 DATA 41,4E,4B,20,4C,4F,4E,43
 225 DATA 5A,45,57,53,4B,49,0D,00
 226 DATA A9,01,A0,01,A2,08,20,BA
 227 DATA FF,A9,0F,A2,D3,A0,02,20
 228 DATA BD,FF,A9,00,85,FB,A9,00
 229 DATA 85,FC,A9,FB,A2,00,A0,00
 230 DATA 4C,D8,FF,48,43,20,45,50
 231 DATA 53,4F,4E,20,20,20,20,20
 232 DATA 20,20,20
```

**Listing der Hardcopy-Routine für Epson-Drucker (Schluß)**

# ESCAPE

## »Escape« für den VC 20 ohne Erweiterung ist ein Spiel in Maschinensprache zum Trainieren des Reaktionsvermögens des Spielers.

Sie müssen versuchen, Ihr Raumschiff möglichst lange vor einer Kollision mit fremden Raumschiffen, Planetoiden und anderen kosmischen Erscheinungen zu retten. Gespielt wird mit einem Joystick. Für den VC 20-Besitzer ohne Joystick kann das Maschinenprogramm leicht durch eine geringfügige Änderung des Programms umgeschrieben werden. Das Eintippen und Abspeichern aller sieben Teilprogramme muß exakt in der Reihenfolge Listing 1 bis 7 erfolgen. Nachdem alle Teilprogramme abgespeichert worden sind, geht man wie folgt vor:

1. Alle Programme bis auf das letzte einladen und starten
2. Falls über Tastatur gespielt werden soll, müssen nun folgende Adressen geändert werden:

POKE 6404,197
POKE 6405,0
POKE 6407,18
POKE 6421,197
POKE 6422,0
POKE 6424,17
POKE 6437,9
POKE 6450,26
POKE 6463,64

statt dem Joystick werden nun folgende Tasten abgefragt: A, W, D, X.

3. Letztes Programm einladen und starten.

Für einen geübten Programmierer läßt sich dieses Programm leicht in nur zwei Teilen (Anleitung und Hauptprogramm) abspeichern (durch ändern der Vektoren 43/44, 45/46 und SAVE"Escape",1,3). Die Programmversion mit den sechs einzelnen Teilprogrammen wurde deshalb gewählt, damit auch ungeübte Programmierer dieses Programm ohne Schwierigkeiten eingeben können.

Da das Maschinenprogramm einen Basicschutz für »Escape-6« enthält, muß dieses »Escape-6« genau eingegeben werden (auch die REM-Zeilen!).

Aber nun viel Spaß beim Spielen. Übrigens beträgt die bisher höchste erreichte Punktzahl 4410. Schaffen Sie mehr?

(Michale Werner/ev)

| | |
|---|---|
| X : | für FOR-NEXT-Schleifen Zeitschleife, |
| A$: | rotierender Text, Bewertung, Ja/Nein-Abfragen |
| B$: | rotierender Titel |
| L : | Länge von A$, Anzahl der verschiedenen Zeichen |
| A : | Länge von B$, zufällige Position |
| D$: | obere Begrenzung des rotierenden Titels |
| E$: | untere Begrenzung des rotierenden Titels |
| C$: | für Anfangen (j/n) |
| F : | Zeitschleife, Anzahl, Hindernisse, Tonhöhe |
| G : | Position der vier Raumbasen |
| M : | jeweilige Zeichen |
| N : | jeweilige Zeichenfarbe |
| F : | Anzahl der zu setzenden Hindernisse |

Benutzte Basic-Variablen und Strings

| | |
|---|---|
| 36879: | Bildschirmfarbe 238 und 25 und 8 (normal 27) |
| 36881: | Bildschirmverschieben 155-38 (normal 38) |
| 56: | Basic-Speicher Ende verschieben 21 (normal 30) |
| 775: | Listschutz 213 (normal 199) |
| 788: | RUN/STOP-Taste ausschalten 194 (normal 191) |
| 37150: | RESTORE-Taste ausschalten 2 (normal 130) |

Liste der verwendeten POKE-Befehle

| | |
|---|---|
| 5380-5470: | Scrollen des Bildschirms nach oben |
| 5471-5561: | Scrollen des Bildschirms nach unten |
| 5562-5658: | Scrollen des Bildschirms nach rechts |
| 5659-5755: | Scrollen des Bildschirms nach links |
| 5756-6066: | Bewegung der Raumschiffe |
| 6067-6097: | Punktewertung |
| 6098-6161: | Setzen neuer Raumschiffe |
| 6162-6192: | Basicprogrammschutz |
| 6193-6272: | Explosion und Herabziehen des Bildschirms |
| 6273-6300: | Bewegung der Hindernisse |
| 6301-6519: | Abfrage d. Joysticks u. Steuerung d. Unterprogramme |
| 6520-6546: | Rotieren des Triebwerkringes |

Speicherplatz des Basicprogramms:

| | |
|---|---|
| 4097-7307: | Basicprogramm zur Punktwertung und Setzen des Bildschirminhaltes zum Spiel |

Speicherplatz der benötigten Sonderzeichen:

| | |
|---|---|
| 7176-7307: | definierte Zeichen für Spiel: |

Benutzte indirekte Adressenspeicher für Maschinensprache:

| | |
|---|---|
| 0-3 | 97-100 |

Speicher für Maschinensprache-»Variablen«:

| | |
|---|---|
| 826-847: | Zwischenspeicher für Zeichen beim Bildschirmscrollen |
| 848-869: | Farbzwischenspeicher für Farbe der Zeichen beim Bildschirmscrollen |
| 870: | augenblickliches Zeichen zur Bewegung der Raumschiffe |
| 871: | neue Flugrichtung der Raumschiffe (1-8) |
| 872: | verloren? wenn Inhalt von 872 = 1, dann Rücksprung ins Basic |
| 873,874,875: | Punktezähler |
| 876,877: | Taktzahl für das Setzen neuer Raumschiffe |
| 878: | Triebwerkring rotieren (1-4) |

Speicherbelegung von »Escape«

```
10 PRINT"ANLEITUNG (J/N)?"
11 GETA$:IFA$="J"THENA=1:GOTO20
12 IFA$="N"THEN20
13 GOTO11
20 PRINT"COPYRIGHT BY
MICHAEL WERNER1983"
21 FORX=1TO3000:NEXT
24 IFA=0THEN55
25 PRINT"SIE HABEN SICH DURCH EIN
EN FEHLER IHRES "
26 PRINT"HYPERRAUMTRIEBWERKESMIT
IHREM RAUMSCHIFF"
28 PRINT"IN EINEM VERKEHRS-REICH
EN SEKTOR DERGALAXIS VERIRRT."
30 PRINT"DIESER SEKTOR WIRD VONEINE
R UNTERENT-WICKELTEN RASSE"
31 PRINT"SPACE":WAIT19
7,32
33 PRINT"BEVOELKERT."
34 PRINT"DEREN RAUMSCHIFFEWERDEN
 MIT VERALTETENATOMTRIEBWERKEN"
35 PRINT"BETRIEBEN. ALSO HUETENSIE
SICH DAVOR MITDIESEN FREMDEN RAUM-"
36 PRINT"SCHIFFEN ZUSAMMENZU-STOSSE
N!"
37 PRINT"SHIFT":WAIT6
54,1
40 PRINT"SIE STEUERN IHRRA
UMSCHIFF MIT JOY-STICK."
41 PRINT"VERSUCHEN SIE MOEG-LICHS
T LANGE DURCH GEKONNTE MANOEVER DEN"
42 PRINT"FREMDEN RAUMSCHIFFEN AUSZUW
EICHEN. PASSENSIE HIERBEI AUCH AUF"
43 PRINT"RETURN":WAIT197
,15
45 PRINT"DIE SICH STAENDIGVERAEN
DERNDEN HINDER-NISSE (MANCHMAL FAST"
46 PRINT"UNSICHTBAR) AUF.AUSSER
DEM KOENNEN SIENICHT UEBER DEN"
47 PRINT"WELTRAUMMUELL,UEBERDIE PLA
NETEN UND UEBERDIE BASEN DER FREMDEN"
48 PRINT"RASSE HINWEGFLIEGEN!"
49 PRINT"SPACE":WAIT197,3
2
50 PRINT"ALSO,CAPTAIN,VER-SUCH
EN SIE IHR GLUECK!ABER ACHTUNG:"
51 PRINT"SOLANGE SIE SICH NICHT BEWE
GEN,BEKOMMEN SIEAUCH KEINE PUNKTE!
"
55 PRINT"NUN BITTE DIE FOLG
ENDEN 6 KURZ-"
56 PRINT"PROGRAMME EINLADEN UND A
UCH STARTEN!!":NEW
READY.
```

Listing 1. Anleitung zu »Escape«

```
SEARCHING FOR $

0 PRINT""
1 CLR:POKE56,21:FORX=0TO511:POKE7168+X,P
EEK(32768+X):NEXT:FORX=0TO131:READA
2 POKE7176+X,A:NEXT
3 DATA0,48,121,251,174,251,121,48,240,18
4,248,232,120,4,3,2,56,108,254,238,124,1
6,56,108
4 DATA15,29,31,23,30,32,192,64,12,158,22
3,117,223,158,12,0,64,192,32,30,23,31,29
,15
5 DATA54,28,8,62,119,127,54,28,2,3,4,120
,232,248,184,240,60,126,255,153,255,126,
189,129
6 DATA136,73,43,31,255,19,246,124,17,146
,212,248,255,200,111,62,58,31,21,250,31,
43,73
7 DATA136,92,248,168,95,248,212,146,17,1
6,20,88,63,252,26,40,8,0,78,124,48,26,30
,12,0
8 DATA44,2,40,128,66,16,5,144,153,51,102
,204
9 FORX=5380TO5561:READA:POKEX,A:NEXT
10 DATA162,0,189,0,30,157,58,3,189,0,150
,157,80,3,232,224,22,208,239,160,24,169,
32,145
11 DATA0,162,0,189,22,30,157,0,30,189,22
,150,157,0,150,232,224,255,208,239,169,9
,145,0
12 DATA169,0,145,2,162,0,189,21,31,157,2
55,30,189,21,151,157,255,150,232,224,251
,208,239
13 DATA162,0,189,58,3,157,228,31,189,80,
3,157,228,151,232,224,22,208,239,96
14 DATA162,0,189,228,31,157,58,3,189,228
,151,157,80,3,232,224,22,208,239,160,24,
169,32
15 DATA145,0,162,255,189,228,30,157,250,
30,189,228,150,157,250,150,202,224,0,208
,239
16 DATA169,9,145,0,169,0,145,2,162,229,1
89,255,29,157,21,30,189,255,149,157,21,1
50,202
17 DATA224,0,208,239,162,0,189,58,3,157,
0,30,189,80,3,157,0,150,232,224,22,208,2
39,96
READY.
```

Listing 2. Grafik und Maschinensprache für »Escape«

```
0 PRINT"BITTE WARTEN!!"
1 POKE56,21:FORX=5562TO5870:READA:POKEX,
A:NEXT
2 DATA169,0,133,97,133,99,169,30,133,98,
169,150,133,100,162,0,160,24,169,32,145,
0,160,21
3 DATA177,97,141,58,3,177,99,141,80,3,13
6,177,97,200,145,97,136,177,99,200,145,9
9,136
4 DATA192,0,208,239,173,58,3,145,97,173,
80,3,145,99,160,24,169,9,145,0,169,0,145
,2,165
5 DATA97,24,105,22,144,4,230,98,230,100,
133,97,165,99,24,105,22,133,99,232,224,2
3,208
6 DATA176,96,169,0,133,97,133,99,169,30,
133,98,169,150,133,100,162,0,160,24,169,
32,145
7 DATA0,160,0,177,97,141,58,3,177,99,141
,80,3,200,177,97,136,145,97,200,177,99,1
36,145
8 DATA99,200,192,21,208,239,173,58,3,145
,97,173,80,3,145,99,160,24,169,9,145,0,1
69,0,145
9 DATA2,165,97,24,105,22,144,4,230,98,23
0,100,133,97,165,99,24,105,22,133,99,232
,224,23
10 DATA208,176,96,169,255,133,97,133,99,
169,29,133,98,169,149,133,100,162,0,160,
1,177
11 DATA97,201,5,16,4,105,4,145,97,200,23
2,224,23,208,240,160,24,177,97,201,1,208
```

```
,7,141
12 DATA102,3,160,23,208,31,201,2,208,7,1
41,102,3,160,1,208,20,201,3,208,7,141,10
2,3,160
13 DATA2,208,9,201,4,208,8,141,102,3,160
,3,32,103,23,165,97,24,105,1,144,4,230,9
8,230
14 DATA100,133,97,165,99,24,105,1,133,99
,165,97,201,226,208,181,165,98,201,31,20
8,175
READY.
```

**Listing 3. Maschinenspracheroutinen für »Escape«**

```
0 PRINT"BITTE WARTEN!!"
1 POKE56,21:FORX=5871TO6161:READA:POKEX,
A:NEXT
2 DATA169,202,133,97,133,99,169,31,133,9
8,169,151,133,100,162,0,160,23,177,97,20
1,9,16,8
3 DATA201,5,48,4,233,4,145,97,200,232,22
4,23,208,236,160,24,177,97,201,5,208,7,1
41,102,3
4 DATA160,25,208,31,201,6,208,7,141,102,
3,160,47,208,20,201,7,208,7,141,102,3,16
0,46,208
5 DATA9,201,8,208,8,141,102,3,160,45,32,
103,23,165,97,56,233,1,176,4,198,98,198,
100,133
6 DATA97,165,99,56,233,1,133,99,165,97,2
01,231,208,181,165,98,201,29,208,175,96,
174,103
7 DATA3,224,8,240,5,238,103,3,208,5,162,
1,142,103,3,177,97,201,32,208,16,173,102
,3,145
8 DATA97,169,2,145,99,160,24,169,32,145,
97,96,201,9,208,8,169,1,141,104,3,24,144
,243,201
9 DATA14,240,13,201,15,240,9,201,16,240,
5,201,9,48,1,96,160,24,173,103,3,145,97,
96
10 DATA173,105,3,24,105,3,144,3,238,106,
3,141,105,3,173,106,3,201,255,240,1,96,1
69,0,141
11 DATA106,3,238,107,3,96,169,100,133,97
,133,99,169,30,133,98,169,150,133,100,17
3,108,3
12 DATA24,105,1,144,3,238,109,3,141,108,
3,173,108,3,201,50,240,1,96,172,36,145,1
77,97
13 DATA201,32,240,1,96,173,103,3,145,97,
169,2,145,99,169,0,141,108,3,141,109,3,9
6
READY.
```

**Listing 4. Maschinenspracheroutinen für »Escape«**

```
0 PRINT"BITTE WARTEN!!"
1 POKE56,21:FORX=6162TO6371:READA:POKEX,
A:NEXT
2 DATA173,100,20,201,0,240,3,76,34,253,1
73,101,20,201,0,240,3,24,144,243,173,102
,20,201
3 DATA0,240,3,24,144,233,96,162,0,160,0,
173,36,145,141,15,144,173,36,145,141,12,
144,201
4 DATA128,48,246,200,192,255,208,235,232
,224,150,208,228,162,191,142,20,3,169,23
8,141
```

```
5 DATA15,144,169,0,133,161,165,161,201,1
,208,250,169,38,141,17,144,24,105,1,201,
180,208
6 DATA1,96,162,0,160,0,200,192,255,208,2
51,232,224,10,208,244,24,144,228
7 DATA234,234,234,110,128,28,46,129,28,1
10,130,28,46,131,28,110,132,28,46,133,28
,110,134
8 DATA28,46,135,28,96,169,0,141,104,3,14
1,105,3,141,106,3,141,107,3,141,108,3,14
1,109,3
9 DATA169,238,141,15,144,169,15,141,14,1
44,169,194,141,20,3,169,2,141,30,145,169
,21
10 DATA133,56,169,1,141,110,3,169,1,141,
103,3,169,229,133,0,133,2,169,30,133,1,1
69
11 DATA150,133,3,32,18,24
READY.
```

**Listing 5. Maschinenspracheroutinen für »Escape«**

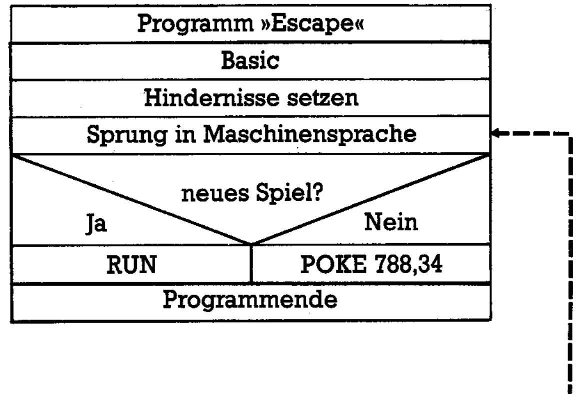
```
0 PRINT"BITTE WARTEN!!"
1 POKE56,21:FORX=6372TO6546:READA:POKEX,
A:NEXT:POKE775,213
2 DATA32,179,23,32,124,22,32,210,23,32,1
29,24,32,120,25,173,104,3,201,1,208,4,32
,49,24
3 DATA96,169,127,141,34,145,173,32,145,2
01,119,208,5,160,25,76,66,25,169,255,141
,34,145
4 DATA173,31,145,201,110,208,5,160,23,76
,66,25,201,46,240,247,201,122,208,5,160,
2,76
5 DATA66,25,201,58,240,247,201,118,208,5
,160,46,76,66,25,201,54,240,247,201,24,2
40,165
6 DATA177,0,201,32,240,7,201,9,176,155,7
6,250,24,169,255,141,12,144,192,23,208,3
,32,186
7 DATA21,192,2,208,3,32,95,21,192,46,208
,3,32,4,21,192,25,208,3,32,27,22,169,0,1
41,12
8 DATA144,76,228,24,174,110,3,224,4,240,
5,238,110,3,208,5,162,1,142,110,3,174,11
0,3,189
9 DATA135,28,141,75,28,96
READY.
```

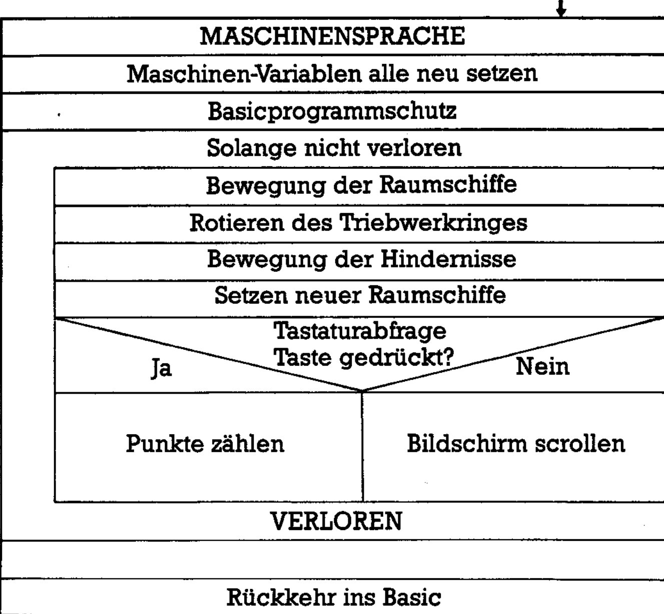
**Listing 6. Maschinenspracheroutinen für »Escape«**

```
0 PRINT"":POKE36879,8:A$=" VON MICHAEL
WERNER ":L=20:B$="     * ESCAPE *     ":A=
18:F=50
1 POKE56,21:RESTORE:D$="_________________
":E$="─────────────────"
2 A$=RIGHT$(A$,1)+LEFT$(A$,L-1):B$=RIGHT
$(B$,A-1)+LEFT$(B$,1)
3 PRINT"";A$;LEFT$(A$,2):PRINT"";D
$;B$;LEFT$(B$,4);E$
4 PRINT"ANFANGEN(J)?":GETC$:IFC
$="J"THEN7
5 PRINT"";A$;LEFT$(A$,2):FORX=1T
OF:NEXT:F=F-1:GOTO2
7 PRINT"":POKE36869,255:POKE36878,15:PO
KE36879,238:POKE7933,9:POKE38653,0
8 READG:IFG>0THENPOKE30720+G,0:POKE30721
+G,0:POKE30742+G,0:POKE30743+G,0
9 IFG>0THENPOKEG,10:POKEG+1,11:POKEG+22,
```

```
12:POKEG+23,13:GOTO8
10 F=0:L=0:M=14:N=7:DATA7721,7815,7962,8
064,0,16,1,15,4
11 A=INT(RND(1)*506)+7680:IFPEEK(A)<>32T
HEN11
12 POKEA,M:POKEA+30720,N:F=F+1:IFF<5THEN
11
13 L=L+1:IFL<3THENREADM:READN:F=0:GOTO11
14 SYS6301:POKE788,194:POKE36869,240
15 F=PEEK(873)+256*PEEK(874)+4096*PEEK(8
75):IFF>2200THENA$="SOGAR":GOTO20
16 IFF>1600THENA$="UNERWARTET":GOTO20
17 IFF>1000THENA$="MUEHEVOLL":GOTO20
18 IFF<500THENA$="NUR":GOTO20
19 A$="GERADE"
20 POKE36879,25:PRINT"[illegible]SIE KONNTEN
";A$
21 PRINT"[illegible]";F;"[illegible]PUNKTE ERREICH
EN!!"
22 PRINT"[illegible]TRAUEN SIE SICH NOCH ?[illegible]
(J/N)"
23 F=128:FORG=155TO38STEP-1:F=F+1:POKE36
876,F:POKE36881,G:NEXT
24 POKE36876,0:GETA$:IFA$="J"THENRUN
25 IFA$="N"THENPOKE788,34
26 GOTO24
27 REM M. WERNER
28 REM PARADEIS 38
29 REM 8120 WEILHEIM
READY.
```

**Listing 7. »Escape« — Das Hauptprogramm**




**Die Programmstruktur von »Escape«**

# Pac-Boy — die Herausforderung

**Die Herausforderung, die dieses Programm an Sie stellt, ist zweifach. Zum ersten sind dies die 128 verschiedenen Schnelligkeitsgrade. Andererseits müssen Sie, um das Programm einzugeben, 5939 — in Worten: fünftausendneunhundertneununddreißig — DATAs eintippen.**

Das Programm lehnt sich stark an die herkömmlichen »Pacman«-Spiele an und ist mit folgenden Einzelheiten ausgestattet (Bilder dazu auf Seite 68):
— Es werden alle acht Sprites, teilweise in Multicolor, verwendet,
— Es wird ein vollständiger neuer Zeichensatz definiert,
— Das Programm läuft mit Ausnahme der Einleitung und des Endes in reiner Maschinensprache, und ist daher sehr schnell,
— High-Scorewertung,
— Einsatz des Soundchips.

**Pogrammbeschreibung:**

Nach bekanntem Schema muß der Spieler bei diesem Spiel versuchen, seinen »Pac-Boy« möglichst lange durch ein Labyrinth zu bewegen, ohne dabei von den umherlaufenden Geistern erwischt zu werden. Frißt man die Pillen, so erhält man jeweils fünf Punkte. Indem man aber eines der vier Nahrungsstücke frißt, kann man kurzzeitig den Spieß umdrehen und die nun flüchteten Geister jagen. Man erhält hier pro erlegtem Geist einen Bonus von 200 Punkten.

Der Zustand der Geister zeigt sich immer in deren Farbe.

**Zur Bedienung des Programmes:**

Zuerst sieht man eine kurze Einleitung, in der alle Mitwirkenden vorgestellt werden. Darauf folgen die Wahl der Schwierigkeitsgrade und das eigentliche Spiel. Gelenkt wird der Vielfraß übrigens mit den Tasten f 5, f 7, z und x für oben, unten links und rechts.

**Zum Eingeben des Programmes:**

Zuerst müssen die beiden Listings »Pac-Boy« und »Data-Pac« eingegeben und dann abgespeichert werden.

Jetzt startet man »Data-Pac« und kann sich aussuchen, ob man die DATA-Zeilen testen oder im Programm fortschreiten will.

Wenn keine Fehler aufgetreten sind, wird dann ein neues, reines Datenprogramm generiert, das dann später sehr schnell vom Hauptprogramm nachgeladen werden kann.

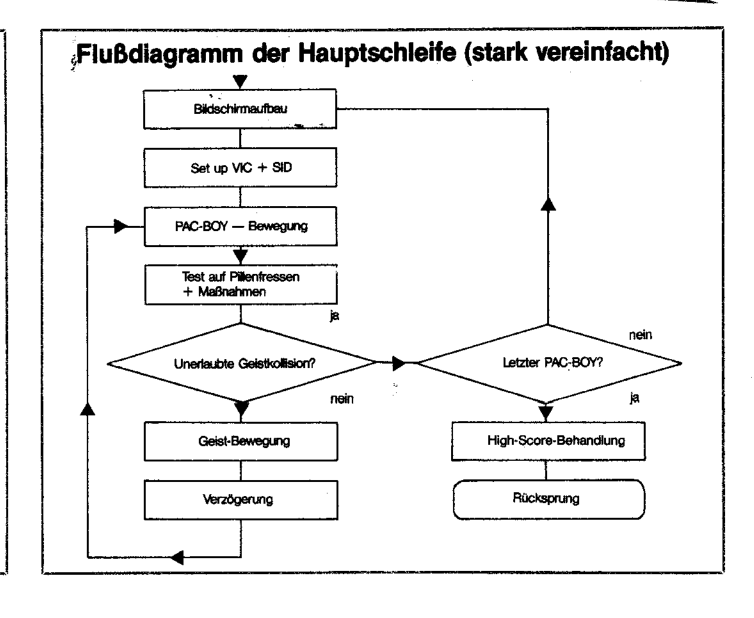
Gestartet wird das Programm duch das Einladen von »Pac-Boy« und nachfolgendem RUN. Zuletzt ist noch zu bemerken, daß die Floppy-Station nach Abänderung der entsprechenden Zeile durch die Datasette ersetzt werden kann.

(H. Schlangmann/rg)

**Für den Interessierten eine Auflistung der Maschinen-Unterprogramme und ein Flußdiagramm:**

| | | |
|---|---|---|
| $ 3000 | → 37FF | Zeichen-DATAs |
| $ 3800 | → 3BE7 | Labyrinth |
| $ 3BE8 | → 3C17 | VIC-Daten |
| $ 3C18 | → 3C1F | Spriteblöcke |
| $ 3C20 | → 3CBF | Zwischenspeicher |
| $ 3CC0 | → 3FFF | Spritedaten (Block 0 - 12) |
| $ 4000 | → 4143 | Tastaturabfrage + Bewegung |
| $ 4144 | → 4176 | Set up VIC, Set up SID |
| $ 417A | → 41D9 | Bildschirmaufbau 1. Teil |
| $ 41DA | → 4330 | Kollisionstests, Sound, Score |
| $ 433C | → 437E | Bildschirmaufbau 2. Teil |
| $ 437F | → 44FD | verschiedene Geisterbewegungsunterprogramme |
| $ 4500 | → 4574 | Hauptschleife (siehe Flußdiagramm) + High-Score |
| $ 458C | → 4731 | Geistbewegung |

**Flußdiagramm der Hauptschleife (stark vereinfacht)**




Listing »PAC-BOY«

```
10 GOTO036:******************************
12 *                                    *
14 * PAC-BOY, EIN ARCADE-GAME           *
16 *                                    *
18 * GESCHRIEBEN VON:                   *
20 *                                    *
22 * HARALD SCHLANGMANN                 *
24 *                                    *
26 * MOOSRAIN 54                        *
28 *                                    *
30 * 8110 MURNAU                        *
32 *                                    *
34 ************************************
36 :
38 :
40 IFSW=0THENSW=1:LOAD"MA-PAC",8,1
42 SYS16708:POKE53248+21,0:RESTORE
44 DATA180,131,180,155,214,155,248,155
46 DATA180,179,214,179,248,179,26,179
47 DATA128
50 FORT=0TO16:READA:POKE53248+T,A:NEXT
52 GOSUB1000
53 POKE53248+21,255:POKE198,0
54 GETA$:IFA$=""THEN54
56 POKE53248+21,0:PRINT"             ** 
** PAC/ BOY ****
58 PRINT
60 PRINT"
        ";
62 PRINT" ZUERST KANNST DU ABER NOCH DI
E     ";
64 PRINT" SCHWIERIGKEITSSTUFE AUS DEM B
EREICH    ";
66 PRINT" Ä0..127Ü WAEHLEN, WOBEI 0 AM
SCHNELL-  ";
68 PRINT" STEN IST...
        ";
70 PRINT"
        ";
72 PRINT
74 PRINT
76 PRINT
78 INPUT "DEINE EINGABE ";SE
80 IFSE<0ORSE>127THENPRINT"
                        ":GOTO78
81 POKE17709,SE
82 PRINT
84 PRINT
86 PRINT
90 PRINT"MIT EINEM TASTENDRUCK GEHTS
LOS...  "
92 GETA$:IFA$=""THEN92
95 FORT=0TO500:NEXT
100 REM ***** HAUPTROUTINE *****
110 SYS17664
120 FORT=0TO1000:NEXT:POKE53248+21,0
130 GOSUB1100
140 GETA$:IFA$<>" "THEN140
150 GOTO42
1000 PRINT"                **** PAC/ ";
1001 PRINT" BOY ****               ";
1002 PRINT"                        ";
1003 PRINT"                        ";
1004 PRINT"                        ";
1005 PRINT"                        ";
1006 PRINT" VERSUCHE MIT DEINEM";
1007 PRINT" 'PAC-BOY' IN EINEM ";
1008 PRINT" LABYRINT MOEGLICHST";
1009 PRINT" LANGE DEN UMHER-    ";
1010 PRINT" LAUFENDEN GEISTERN ";
1011 PRINT"AUSZUWEICHEN UND    ";
1012 PRINT" DABEI VIEL ZU FRESS";
1013 PRINT"EN...               ";
1014 PRINT"                    ";
1015 PRINT"                    ";
1016 PRINT"                    ";
1017 PRINT"                    ";
1018 PRINT"DIE SPIELFIGUREN:  ";
1019 PRINT"                    ";
1020 PRINT"                    ";
1021 PRINT"                    ";
1022 PRINT"          DER HELD : ";
1023 PRINT"                    ";
1024 PRINT"                    ";
1025 PRINT"                    ";
1026 PRINT"                    ";
1027 PRINT"                    ";
1028 PRINT" DIE DREI GEISTER : ";
1029 PRINT"                    ";
1030 PRINT"                    ";
1031 PRINT"                    ";
1032 PRINT"                    ";
1033 PRINT"                    ";
1034 PRINT"        DIE NAHRUNG : ";
1035 PRINT"                    ";
1036 PRINT"                    ";
1037 PRINT"                    ";
```

```
1038 PRINT"                              ";
1039 PRINT"                              ";
1040 PRINT" UND VIELE, VIELE PI";
1041 PRINT"LLEN :■ ● ● ● ● ● ● ⊠ ";
1042 PRINT"                              ";
1043 PRINT"                              ";
1044 PRINT"                              ";
1045 PRINT"                              ";
1046 PRINT" ─■*** DRUECKE ZUM ST■";
1047 PRINT"■ART EINE TASTE ***■─ ";
1048 PRINT"                              ";
1099 RETURN
1100 PRINT"■■             ─■**** PAC/■";
1101 PRINT"■ BOY ****■─            ";
1102 PRINT"                              ";
1103 PRINT"                              ";
1104 PRINT"                              ";
1105 PRINT"                              ";
1106 PRINT"■ ■  ■  ■  ■  ─  ─  ■  ■ ";
1107 PRINT" ■  ■  ─  ─  ■  ■  ■  ■";
1108 PRINT"■ ■|L─ ■ ■|L■ ■ ■ ■|\■ ■ ■ ■|
 ";
1109 PRINT" ■ ■|L■ ■ ■ ■  ■ ■ ■ ■|L─ ■ ■
|L■ ■";
1110 PRINT"■ ■     ■ ■|\■ ■ ■     ■ ■ ■
 ";
1111 PRINT" ■ ■  ■ ■ ■ ■  ■ ■ ■ ■     ■ ■
|\■ ■";
1112 PRINT"■ ■  ⌒ ■     ■ ■ ■∪■ ■ ■    ■─
 ";
1113 PRINT" ■ ■  ■ ■ ■ ■  ■ ■ ■    ■─ ■
■";
1114 PRINT"■ ■  ■ ■ ■ ■|L■ ■ ■ ■   ■ ■ ■
■    ";
1115 PRINT" ■ ■  ■ ■ ■ ■  ■ ■ ■ ■     ■
─■ ";
1116 PRINT"■ ■|\■ ■ ■ ■   ■ ■ ■ ■   ■ ■ ■
■|\─ ";
1117 PRINT" ■ ■|\■ ■  ■■■■■  ■ ■|\─ ■ ■
■ |";
1118 PRINT" ■  ■  ─  ─  ─  ─  ■  ■  ";
1119 PRINT" ■  ■   ∪   ■  ■  ─ ■─";
1120 PRINT"                              ";
1121 PRINT"                              ";
1122 PRINT"                              ";
1123 PRINT"                              ";
1124 PRINT"                              ";
1125 PRINT"                              ";
1126 PRINT"                              ";
1127 PRINT"                              ";
1128 PRINT"■ ■                       ■";
1129 PRINT"■                      ■  ";
1130 PRINT" ■ DER GEIST WAR WOHL■";
1131 PRINT"■ SCHNELLER...          ■ ";
1132 PRINT" ■ WENN DU DENN MUT N■";
1133 PRINT"■OCH NICHT VERLOREN ■ ";
1134 PRINT" ■ HAST DANN DRUECKE ■";
1135 PRINT"■FUER EIN WEITERES  ■ ";
1136 PRINT" ■ SPIEL DIE 'SPACE'-■";
1137 PRINT"■TASTE...              ■ ";
1138 PRINT" ■                     ■";
1139 PRINT"■                    ■  ";
1140 PRINT"                              ";
1141 PRINT"                              ";
1142 PRINT"                              ";
1143 PRINT"                              ";
1144 PRINT"■ ─■ICH WARTE AUF EINE■";
1145 PRINT"■ EINGABE...■─          ";
1150 RETURN
1151 DATA47
READY.
```

**Listing »DATA-PAC«**

```
0 DATA60,102,110,110,96,98,60,0,60,34,34
,62,98,98,98,0,120,100,100,124
1 DATA98,98,126,0,62,70,64,64,64,66,62,0
,124,102,98,98,98,98,126,0,126
2 DATA64,96,120,96,96,126,0,126,96,96,56
,32,32,32,0,126,102,96,108,110
3 DATA98,126,0,68,68,68,126,102,102,102,
0,8,8,24,24,24,24,24,0,4,4,4,6
4 DATA102,102,24,0,68,72,112,112,120,108
,102,0,64,64,96,96,96,102,126,0
5 DATA102,90,66,66,98,114,114,0,100,84,7
6,68,102,102,102,0,60,102,66,70
6 DATA70,110,60,0,124,70,66,124,96,96,96
,0,60,102,66,90,94,110,60,0,120
7 DATA76,68,124,104,102,98,0,60,70,64,60
,6,70,60,0,124,20,16,24,24,24,24
8 DATA0,66,66,66,70,70,70,62,0,70,70,78,
78,76,40,16,0,102,102,98,98,90
9 DATA126,36,0,66,68,40,16,40,76,78,0,70
,70,70,36,24,24,24,0,126,98,4,8
10 DATA16,54,126,0,60,48,48,48,48,48,60,
0,12,18,48,124,48,98,252,0,60,12
11 DATA12,12,12,12,60,0,16,56,124,16,24,
24,24,0,0,32,111,255,96,32,0,0,0
12 DATA0,0,0,0,0,0,56,56,24,24,0,24,24
,0,102,102,34,0,0,0,0,0,36,36,126
13 DATA36,126,36,36,0,24,62,96,60,6,124,
24,0,99,102,12,24,48,115,115,0,60
14 DATA66,36,56,103,98,63,0,4,8,24,0,0,0
,0,0,4,8,16,16,24,12,4,0,16,8,4
15 DATA4,12,24,16,0,0,66,52,255,52,66,0,
0,0,16,16,126,24,24,0,0,0,0,0,0
16 DATA0,24,24,48,0,0,0,126,14,0,0,0,0,0
,0,0,0,24,24,0,0,2,4,8,24,48,96
17 DATA0,60,70,74,82,114,114,60,0,8,24,4
0,8,24,24,126,0,60,98,2,12,48,112
18 DATA126,0,60,98,2,28,6,102,60,0,6,10,
18,34,127,6,6,0,126,64,124,2,6,102
19 DATA60,0,60,70,64,124,102,102,60,0,12
6,98,4,8,24,24,24,0,60,66,66,60
20 DATA98,102,60,0,60,66,66,62,6,102,60,
0,0,0,24,0,0,24,0,0,0,0,24,0,0,24
21 DATA24,48,8,16,32,96,48,24,8,0,0,0,12
6,0,126,0,0,0,16,8,4,6,12,24,16
22 DATA0,60,98,2,12,24,0,24,0,0,0,0,255,
255,0,0,0,0,0,0,0,1,7,31,255,0,0
23 DATA0,0,128,224,248,255,31,127,255,25
5,255,255,127,31,248,254,255,255
24 DATA255,255,254,248,60,126,126,255,25
5,255,255,255,255,255,255,255,255
25 DATA126,126,60,224,192,128,0,0,0,0,0,
0,0,0,0,0,128,192,224,240,252,254
26 DATA254,255,255,255,255,255,255,255,2
55,127,127,63,15,255,255,255,255
27 DATA254,254,252,240,7,3,1,0,0,0,0,0,0
,0,0,0,0,1,3,7,0,0,0,7,7,0,0,0,0
28 DATA0,0,224,224,0,0,0,255,255,3,3,3,3
,3,3,0,60,126,126,126,126,60,0,0
29 DATA0,0,0,0,255,255,0,54,127,127,127,
62,28,8,0,96,96,96,96,96,96,96,96
30 DATA15,63,127,127,255,255,255,255,195
,231,126,60,60,126,231,195,0,60
31 DATA126,102,102,126,60,0,24,24,102,10
2,24,24,60,0,6,6,6,6,6,6,6,6,8,28
32 DATA62,127,62,28,8,0,24,24,24,255,255
,24,24,24,192,192,48,48,192,192
33 DATA48,48,24,24,24,24,24,24,24,24,0,0
,3,62,118,54,54,0,255,127,63,31
34 DATA15,7,3,1,0,0,0,0,0,0,0,0,240,240,
```

**Listing »DATA-PAC« (Fortsetzung)**

```
240,240,240,240,240,240,0,0,0,0
35 DATA255,255,255,255,255,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,255,192,192,192,192
36 DATA192,192,192,192,204,204,51,51,204
,204,51,51,3,3,3,3,3,3,3,3,0,0,0
37 DATA0,204,204,51,51,255,254,252,248,2
40,224,192,128,3,3,3,3,3,3,3,3,24
38 DATA24,24,31,31,24,24,24,0,0,0,0,15,1
5,15,15,24,24,24,31,31,0,0,0,0,0
39 DATA0,248,248,24,24,24,0,0,0,0,0,0,25
5,255,0,0,0,31,31,24,24,24,24,24
40 DATA24,255,255,0,0,0,0,0,0,255,255,24
,24,24,24,24,24,248,248,24,24,24
41 DATA192,192,192,192,192,192,192,192,2
24,224,224,224,224,224,224,224,7
42 DATA7,7,7,7,7,7,7,255,255,0,0,0,0,0,0
,255,255,255,0,0,0,0,0,0,0,0,0,0
43 DATA255,255,255,3,3,3,3,3,3,255,255,0
,0,0,0,240,240,240,240,15,15,15
44 DATA15,0,0,0,0,24,24,24,248,248,0,0,0
,240,240,240,240,0,0,0,0,240,240
45 DATA240,240,15,15,15,15,195,153,145,1
45,159,157,195,255,195,221,221,193
46 DATA157,157,157,255,135,155,155,131,1
57,157,129,255,193,185,191,191,191
47 DATA189,193,255,131,153,157,157,157,1
57,129,255,129,191,159,135,159,159
48 DATA129,255,129,159,159,199,223,223,2
23,255,129,153,159,147,145,157,129
49 DATA255,187,187,187,129,153,153,153,2
55,247,247,231,231,231,231,231,255
50 DATA251,251,251,249,153,153,231,255,1
87,183,143,143,135,147,153,255,191
51 DATA191,159,159,159,153,129,255,153,1
65,189,189,157,141,141,255,155,171
52 DATA179,187,153,153,153,255,195,153,1
89,185,185,145,195,255,131,185,189
53 DATA131,159,159,159,255,195,153,189,1
65,161,145,195,255,135,179,187,131
54 DATA151,153,157,255,195,185,191,195,2
49,185,195,255,131,235,239,231,231
55 DATA231,231,255,189,189,189,185,185,1
85,193,255,185,185,177,177,179,215
56 DATA239,255,153,153,157,157,165,129,2
19,255,189,187,215,239,215,179,177
57 DATA255,185,185,185,219,231,231,231,2
55,129,157,251,247,239,201,129,255
58 DATA195,207,207,207,207,207,195,255,2
43,237,207,131,207,157,3,255,195
59 DATA243,243,243,243,243,195,255,239,1
99,131,239,231,231,231,255,255,223
60 DATA144,0,159,223,255,255,255,255,255
,255,255,255,255,255,199,199,231
61 DATA231,255,231,231,255,153,153,221,2
55,255,255,255,255,219,219,129,219
62 DATA129,219,219,255,231,193,159,195,2
49,131,231,255,156,153,243,231,207
63 DATA140,140,255,195,189,219,199,152,1
57,192,255,251,247,231,255,255,255
64 DATA255,255,251,247,239,239,231,243,2
51,255,239,247,251,251,243,231,239
65 DATA255,255,189,203,0,203,189,255,255
,255,239,239,129,231,231,255,255
66 DATA255,255,255,255,255,231,231,207,2
55,255,255,129,241,255,255,255,255
67 DATA255,255,255,255,231,231,255,255,2
53,251,247,231,207,159,255,195,185
68 DATA181,173,141,141,195,255,247,231,2
15,247,231,231,129,255,195,157,253
69 DATA243,207,143,129,255,195,157,253,2
27,249,153,195,255,249,245,237,221
70 DATA128,249,249,255,129,191,131,253,2
49,153,195,255,195,185,191,131,153
71 DATA153,195,255,129,157,251,247,231,2
31,231,255,195,189,189,195,157,153
72 DATA195,255,195,189,189,193,249,153,1
95,255,255,255,231,255,255,231,255
73 DATA255,255,255,231,255,255,231,231,2
07,247,239,223,159,207,231,247,255
74 DATA255,255,129,255,129,255,255,255,2
39,247,251,249,243,231,239,255,195
75 DATA157,253,243,231,255,231,255,255,2
55,255,0,0,255,255,255,255,255,255
76 DATA255,254,248,224,0,255,255,255,255
,127,31,7,0,224,128,0,0,0,0,128
77 DATA224,7,1,0,0,0,0,1,7,195,129,129,0
,0,0,0,0,0,0,0,0,0,129,129,195,31
78 DATA63,127,255,255,255,255,255,255,25
5,255,255,255,127,63,31,15,3,1,1
79 DATA0,0,0,0,0,0,0,0,128,128,192,240,0
,0,0,0,1,1,3,15,248,252,254,255
80 DATA255,255,255,255,255,255,255,255,2
55,254,252,248,255,255,255,248,248
81 DATA255,255,255,255,255,255,31,31,255
,255,255,0,0,252,252,252,252,252
82 DATA252,255,195,129,129,129,129,195,2
55,255,255,255,255,255,0,0,255,201
83 DATA128,128,128,193,227,247,255,159,1
59,159,159,159,159,159,159,240,192
84 DATA128,128,0,0,0,0,60,24,129,195,195
,129,24,60,255,195,129,153,153,129
85 DATA195,255,231,231,153,153,231,231,1
95,255,249,249,249,249,249,249,249
86 DATA249,247,227,193,128,193,227,247,2
55,231,231,231,0,0,231,231,231,63
87 DATA63,207,207,63,63,207,207,231,231,
231,231,231,231,231,231,255,255
88 DATA252,193,137,201,201,255,0,128,192
,224,240,248,252,254,255,255,255
89 DATA255,255,255,255,255,15,15,15,15,1
5,15,15,15,255,255,255,255,0,0,0
90 DATA0,0,255,255,255,255,255,255,255,2
55,255,255,255,255,255,255,0,63
91 DATA63,63,63,63,63,63,63,51,51,204,20
4,51,51,204,204,252,252,252,252
92 DATA252,252,252,252,255,255,255,255,5
1,51,204,204,0,1,3,7,15,31,63,127
93 DATA252,252,252,252,252,252,252,252,2
31,231,231,224,224,231,231,231,255
94 DATA255,255,255,240,240,240,240,231,2
31,231,224,224,255,255,255,255,255
95 DATA255,7,7,231,231,231,255,255,255,2
55,255,255,0,0,255,255,255,224,224
96 DATA231,231,231,231,231,231,0,0,255,2
55,255,255,255,255,0,0,231,231,231
97 DATA231,231,231,7,7,231,231,231,63,63
,63,63,63,63,63,63,31,31,31,31,31
98 DATA31,31,31,248,248,248,248,248,248,
248,248,0,0,255,255,255,255,255
99 DATA255,0,0,0,255,255,255,255,255,255
,255,255,255,255,0,0,0,252,252,252
100 DATA252,252,252,0,0,255,255,255,255,
15,15,15,15,240,240,240,240,255,255
101 DATA255,255,231,231,231,7,7,255,255,
255,15,15,15,15,255,255,255,255,15
102 DATA15,15,15,240,240,240,240,32,32,3
2,32,32,32,32,32,32,32,65,85,170
103 DATA170,170,160,144,129,131,206,207,
130,143,153,160,170,170,170,73,66
104 DATA32,32,32,32,32,32,32,32,32,32,85
,160,160,160,160,160,160,160,160
```

**Listing »DATA-PAC« (Fortsetzung)**

```
105 DATA160,160,160,160,160,160,160,160,
160,160,160,160,160,160,160,160,160
106 DATA160,160,160,160,160,160,160,160,
160,160,160,160,160,73,160,71,32
107 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,32,76,160,160,71,32,32
108 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,76,160,160,32,81,32,32
109 DATA81,32,32,32,32,32,32,32,81,32,32
,32,81,32,160,160,32,81,32,32,32
110 DATA81,32,32,32,81,32,32,32,81,32,32
,81,32,160,160,32,32,32,32,32,32
111 DATA32,32,32,32,32,32,32,32,32,32,32
,32,160,160,32,32,32,32,32,32,32
112 DATA32,32,32,32,32,32,32,32,32,32,32
,160,160,32,32,32,67,160,160,68,32
113 DATA32,32,67,160,160,160,68,32,32,32
,74,75,32,32,32,67,160,160,160,68
114 DATA32,32,32,67,160,160,68,32,32,32,
160,160,32,32,32,32,32,32,32,32,32
115 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,32,32,32,32,32,32,32,32
116 DATA32,32,32,32,32,32,32,160,160,32,
81,32,32,81,32,32,32,81,32,32,32
117 DATA81,32,32,32,81,32,32,32,32,81,32
,32,32,81,32,32,32,32,32,32,32,81
118 DATA32,32,81,32,160,160,72,32,32,32,
32,32,32,32,32,32,32,32,32,32,32
119 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,32,32,32,32,32,32,32,32
120 DATA77,160,74,160,160,160,160,160,16
0,73,32,32,32,69,32,32,32,67,160
121 DATA160,160,160,160,160,160,160,68,3
2,32,32,69,32,32,32,85,160,160,160
122 DATA160,160,160,75,32,32,32,32,32,76
,160,75,32,32,32,160,32,81,32,32
123 DATA32,32,32,32,32,32,32,32,32,32,81
,32,160,32,32,32,74,160,71,32,32
124 DATA32,32,32,32,32,32,32,32,77,160,3
2,32,32,32,70,32,32,32,32,81,32,32
125 DATA32,32,32,32,81,32,32,32,32,70,32
,32,32,76,160,72,32,32,32,32,32,160
126 DATA160,160,160,160,160,75,32,32,32,
32,32,32,32,32,32,32,32,32,32,32
127 DATA32,32,32,32,32,32,32,32,32,32,32
,32,74,160,160,160,160,160,160,32
128 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,85,160,68,32,32,32,32,67
129 DATA160,73,32,32,32,32,32,32,32,32,3
2,32,32,32,32,32,32,32,32,32,32,32
130 DATA32,32,32,32,32,32,32,81,32,32,16
0,71,32,32,32,32,32,32,76,160,32
131 DATA32,81,32,32,32,81,32,32,32,32,32
,32,32,32,32,32,32,32,32,32,32,32
132 DATA32,77,69,32,32,32,77,160,32,32,3
2,32,32,32,32,32,160,32,32,32,32
133 DATA69,72,32,32,32,32,32,32,32,32,32
,160,160,160,160,160,68,32,32,32
134 DATA85,75,32,32,32,67,75,32,32,32,32
,32,32,32,32,74,68,32,32,32,74,73
135 DATA32,32,32,67,160,160,160,160,160,
160,71,32,32,32,32,32,32,32,160,71
136 DATA32,32,32,32,32,32,32,32,85,73,32
,32,32,32,32,32,32,32,76,160,32,32
137 DATA32,32,32,32,32,76,160,160,32,81,
32,32,32,32,81,32,160,32,81,32,32
138 DATA32,32,32,32,32,160,160,32,32,32,
32,32,32,32,81,32,160,32,81,32,32
139 DATA32,32,81,32,160,160,32,32,32,32,
32,32,32,32,160,32,32,32,32,32,32
140 DATA32,32,77,160,160,72,32,32,32,32,
32,32,32,32,160,32,32,32,32,32,32
141 DATA32,32,160,160,32,32,32,67,68,32,
32,32,70,32,32,32,67,160,160,160
142 DATA160,160,160,160,160,160,160,160,
160,68,32,32,32,70,32,32,32,67,68
143 DATA32,32,32,160,160,32,32,32,32,32,
32,32,32,32,32,32,32,32,32,32,32
144 DATA32,32,32,32,32,32,32,32,32,32.32
,32,32,32,32,32,32,32,32,32,32.32
145 DATA160,160,32,81,32,32,32,32,32,32,
32,32,81,32,32,32.81,32,32,81,32
146 DATA32,81.32,32,81,32,32,32,81,32,32
.32,32,32,32,32,32,81,32,160,160
147 DATA72,32,32,32,32,32,32,32,32,32,32
,32,32,32,32,32,32,32,32,32,32,32
148 DATA32,32,32,32,32,32,32,32,32,32,32
,32,32,32,32,77,160,74,160,136,137
149 DATA135,136,173,147,131,143,146,133,
186,176,176,176,176,176,160,160,160
150 DATA147,131,143,146,133,186,176,176,
176,176,176,160,160,160,179,160,149
151 DATA144,75,88,68,152,188,192,188,168
,164,16,212,72,220,0,100,88,148,80
152 DATA27,28,209,0,255,200,0,28,113,240
,0,240,0,0,0,6,6,6,6,6,2,7,8,3,10
153 DATA4,5,5,10,8,0,249,251,251,251,252
,253,254,255,39,74,160,136,137,135
154 DATA136,173,147,131,143,146,133,186,
176,178,176,179,176,160,160,160,147
155 DATA131,143,146,133,186,176,179,180,
182,176,160,160,160,177,160,149,144
156 DATA75,81,0,1,2,40,41,42,80,81,82,0,
0,0,0,0,0,240,240,240,240,15,15,15
157 DATA15,195,153,145,145,159,153,195,2
55,231,195,153,129,153,153,153,255
158 DATA131,153,153,131,153,153,131,255,
195,153,159,159,159,153,195,255,135
159 DATA147,153,153,153,147,135,255,129,
159,159,135,159,159,129,255,129,159
160 DATA159,135,159,159,159,255,195,153,
159,145,153,153,195,255,153,153,153
161 DATA129,153,153,153,255,195,231,231,
231,231,231,195,255,225,243,243,243
162 DATA243,147,199,255,153,147,135,143,
135,147,153,0,0,0,255,0,3,255,192
163 DATA7,255,224,15,255,240,31,255,248,
31,255,248,63,255,252,63,255,252
164 DATA63,255,252,63,231,60,63,231,60,6
3,231,252,63,195,252,31,195,248,31
165 DATA129,248,15,129,240,7,0,224,3,0,1
92,0,0,0,0,0,0,255,0,0,0,0,255,0
166 DATA3,255,192,7,255,224,15,255,240,3
1,255,248,31,255,248,63,255,252,63
167 DATA255,252,63,255,252,63,255,60,63,
255,60,63,239,252,63,239,252,31,239
168 DATA248,31,231,248,15,231,240,7,231,
224,3,231,192,0,231,0,0,0,0,0,0,0
169 DATA0,0,255,0,3,255,192,7,255,224,15
,255,240,31,243,248,31,243,248,63
170 DATA255,224,63,255,0,63,248,0,63,224
,0,63,248,0,63,255,0,63,255,224,31
171 DATA255,248,31,255,248,15,255,240,7,
255,224,3,255,192,0,255,0,0,0,0,0
172 DATA0,0,0,0,255,0,3,255,192,7,255,22
4,15,255,240,31,243,248,31,243,248
173 DATA63,255,252,63,255,252,63,248,0,6
3,224,0,63,255,252,63,255,252,63
174 DATA255,252,31,255,248,31,255,248,15
,255,240,7,255,224,3,255,192,0,255
175 DATA0,0,0,0,255,0,0,0,0,0,0,3,0,192,
```

## Listing »DATA-PAC« (Fortsetzung)

```
7,0,224,15,129,240,31,129,248,31
176 DATA195,248,63,195,252,63,231,252,60
,231,252,60,231,252,63,255,252,63
177 DATA255,252,63,255,252,31,255,248,31
,255,248,15,255,240,7,255,224,3,255
178 DATA192,0,255,0,0,0,0,255,0,0,0,0,19
9,0,3,199,192,7,199,224,15,199,240
179 DATA31,199,248,31,199,248,63,231,252
,63,231,252,60,231,252,60,231,252
180 DATA63,255,252,63,255,252,63,255,252
,31,255,248,31,255,248,15,255,240
181 DATA7,255,224,3,255,192,0,255,0,0,0,
0,0,0,0,0,0,255,0,3,255,192,7,255
182 DATA224,15,255,240,31,207,248,31,207
,248,7,255,252,0,255,252,0,31,252
183 DATA0,7,252,0,31,252,0,255,252,7,255
,252,31,255,248,31,255,248,15,255
184 DATA240,7,255,224,3,255,192,0,255,0,
0,0,0,0,0,0,0,0,255,0,3,255,192,7
185 DATA255,224,15,255,240,31,207,248,31
,207,248,63,255,252,63,255,252,0
186 DATA127,252,0,7,252,63,255,252,63,25
5,252,63,255,252,31,255,248,31,255
187 DATA248,15,255,240,7,255,224,3,255,1
92,0,255,0,0,0,0,255,0,0,0,0,126
188 DATA0,1,255,128,7,255,224,15,255,240
,31,255,248,28,24,56,59,231,220,58
189 DATA36,92,58,165,92,58,36,92,59,231,
220,60,24,60,63,255,252,63,255,252
190 DATA63,255,252,63,255,252,57,231,156
,49,195,140,33,129,132,0,0,0,255
191 DATA0,0,0,0,0,0,0,32,0,2,128,0,42,
32,0,168,40,0,136,168,2,136,40,2
192 DATA10,8,2,2,128,2,2,128,2,0,128,5,0
,128,31,64,80,31,65,116,23,65,244
193 DATA21,65,212,21,65,84,5,1,84,0,0,80
,0,0,0,0,2,128,0,10,160,170,2,162
194 DATA160,0,10,40,0,8,42,0,40,42,0,255
,10,3,255,192,15,255,240,15,255,240
195 DATA63,255,252,63,255,252,63,253,252
,63,245,124,61,117,124,61,125,252
196 DATA15,127,240,15,255,240,3,251,192,
0,235,0,0,40,0,0,0,0,0,16,0,4,16
197 DATA0,4,16,0,4,16,0,4,16,0,4,252,0,6
3,252,60,63,48,60,12,42,170,168,37
198 DATA150,88,37,150,88,37,150,88,37,15
0,88,37,150,88,37,150,88,37,150,88
199 DATA37,150,88,42,170,168,63,255,252,
255,255,255,255,0,0,0,0,0,0,0,0
200 DATA0,0,0,0,12,16,0,63,84,0,63,84,0,
63,84,0,45,84,0,41,95,0,170,95,0
201 DATA170,255,2,170,191,2,170,172,10,1
70,160,10,170,0,42,160,0,42,0,0,168
202 DATA0,0,160,0,0,128,0,0,255,216,234,
234,234,234,165,203,201,3,208,3,76
203 DATA63,64,234,234,201,6,208,3,76,119
,64,234,234,201,12,208,3,76,35,65
204 DATA234,234,201,23,208,3,76,6,65,234
,96,234,76,152,66,232,169,0,56,42
205 DATA202,208,252,45,31,208,234,234,96
,234,234,173,1,208,24,105,8,141,1
206 DATA208,32,102,64,162,0,32,44,64,240
,9,173,1,208,56,233,8,141,1,208,169
207 DATA243,141,248,7,234,234,234,234,23
4,96,234,234,162,9,160,0,234,234
208 DATA136,208,251,202,208,246,234,96,2
34,234,173,1,208,56,233,8,141,1,208
209 DATA32,102,64,162,0,32,44,64,240,9,1
73,1,208,24,105,8,141,1,208,169,247
210 DATA141,248,7,234,234,234,234,234,23
4,234,96,234,234,234,134,20,169,208
211 DATA133,21,6.20,234,234,232,169,0,56
,42,202,208,252,234,133,182,234,234
212 DATA160,0,177,20,24,105,8,145,20,176
,1,96,173,16,208,5,182,141,16,208
213 DATA96,234,234,134,20,169,208,133,21
,6,20,234,232,169,0,56,42,202,208
214 DATA252,234,133,182,56,169,255,229,1
82,133,182,234,160,0,177,20,56,233
215 DATA8,145,20,144,1,96,234,173,16,208
,37,182,141,16,208,96,234,234,162
216 DATA0,32,161,64,32,102,64,162,0,32,4
4,64,240,3,32,208,64,169,245,141
217 DATA248,7,162,0,32,156,67,96,234,234
,162,0,32,208,64,32,102,64,162,0
218 DATA32,44,64,240,5,162,0,32,161,64,1
69,249,141,248,7,169,0,32,214,67
219 DATA96,234,162,46,189,232,59,157,0,2
08,202,16,247,162,7,189,24,60,157
220 DATA248,7,202,16,247,234,234,96,234,
166,162,189,0,160,41,3,96,162,24
221 DATA169,0,157,0,212,202,16,250,169,1
5,141,24,212,96,234,234,234,169,0
222 DATA133,20,133,87,169,4,133,21,169,5
6,133,88,162,24,160,39,177,87,145
223 DATA20,136,16,249,24,169,40,101,87,1
33,87,169,0,101,88,133,88,24,169
224 DATA40,101,20,133,20,169,0,101,21,13
3,21,202,16,218,234,234,234,234,169
225 DATA0,133,20,169,216,133,21,162,24,1
69,14,160,39,145,20,136,16,251,24
226 DATA165,20,105,40,133,20,165,21,105,
0,133,21,202,16,231,76,60,67,162
227 DATA31,189,192,7,24,105,5,157,192,7,
201,181,208,1,96,169,176,157,192
228 DATA7,202,254,192,7,169,186,221,192,
7,240,240,96,234,234,134,2,138,10
229 DATA170,232,173,1,208,221,0,208,240,
1,96,202,173,0,208,221,0,208,240
230 DATA1,96,169,0,166,2,232,56,42,202,2
08,252,24,105,1,133,2,173,16,208
231 DATA37,2,201,0,240,5,197,2,240,1.96,
169,0,96,234,234,169,100,141,0,212
232 DATA169,20,141,1,212,169,129,141,4,2
12,169,15,141,5,212,169,242,141,6
233 DATA212,32,102,64,32,102,64,169,128,
141,4,212,96,162,7,138,72,32,251
234 DATA65,208,42,238,248,7,32,54,66,104
,10,170,232,169,0,157,0,208,32,40
235 DATA68,32,218,65,169,2,32,102,64,56,
233,1,208,248,206,248,7,234,234,234
236 DATA169,0,76,151,66,104,170,202,201,
4,208,200,96,234,234,234,173,16,208
237 DATA41,1,133,21,173,0,208,56,233,16,
133,20,165,21,233,0,133,21,234,70
238 DATA21,102,20,70,21,102,20,70,21,102
,20,24,169,4,101,21,133,21,173,1
239 DATA208,56,233,44,133,92,70,92,70,92
,70,92,162,40,24,165,92,101,20,133
240 DATA20,169,0,101,21,133,21,202,208,2
40,165,20,56,233,41,133,20,165,21
241 DATA233,0,133,21,160,0,169,81,209,20
,240,45,200,209,20,240,40,200,209
242 DATA20,240,35,160,40,209,20,240,29,2
00,209,20,240,24,200,209,20,240,19
243 DATA160,80,209,20,240,13,200,209,20,
240,8,200,209,20,240,3,76,47,64,169
244 DATA32,145,20,238,248,7,32,54,66,206
,32,60,208,3,32,128,67,76,120,66
245 DATA234,169,4,133,21,169,0,133,20,13
3,87,169,216,133,88,234,162,24,160
246 DATA39,177,20,201,81,208,4,169,0,145
```

```
,87,136,16,243,24,165,20,105,40,133
247 DATA20,165,21,105,0,133,21,24,165,87
,105,40,133,87,165,88,105,0,133,88
248 DATA202,16,212,169,39,76,59,68,234,2
34,234,234,162,39,189,192,7,157,33
249 DATA60,202,16,247,32,122,65,162,39,1
89,33,60,157,192,7,202,16,247,96
250 DATA234,134,2,6,2,164,2,200,185,0,20
8,201,156,240,1,96,136,185,0,208
251 DATA201,80,240,1,96,232,169,0,56,42,
202,208,252,45,16,208,208,1,96,133
252 DATA2,169,255,56,229,2,45,16,208,141
,16,208,169,16,153,0,208,96,234,234
253 DATA134,2,6,2,164,2,200,185,0,208,20
1,156,240,1,96,136,185,0,208,201
254 DATA8,240,1,96,232,169,0,56,42,202,2
08,252,133,2,45,16,208,240,1,96,165
255 DATA2,234,234,234,13,16,208,141,16,2
08,169,72,153,0,208,96,234,206,73
256 DATA60,240,1,96,169,3,141,40,208,169
,10,141,41,208,169,4,141,42,208,96
257 DATA234,234,169,5,141,40,208,141,41,
208,141,42,208,169,64,141,73,60,96
258 DATA234,141,32,60,162,7,189,240,59,1
57,8,208,202,16,247,234,96,234,169
259 DATA15,141,5,212,169,251,141,6,212,1
69,17,141,4,212,169,0,133,162,162
260 DATA20,138,72,165,162,141,1,212,32,1
02,64,32,102,64,32,102,64,104,170
261 DATA202,208,235,169,10,141,1,212,169
,16,141,4,212,234,234,96,234,234
262 DATA162,3,134,97,32,251,65,240,8,166
,97,202,208,244,169,255,96,32,76
263 DATA68,173,40,208,201,245,240,3,169,
0,96,166,97,6,97,164,97,185,232,59
264 DATA153,0,208,185,233,59,153,1,208,2
32,56,169,0,42,202,208,252,133,97
265 DATA169,255,56,229,97,45,16,208,141,
16,208,234,234,160,40,32,218,65,32
266 DATA219,68,136,208,247,76,148,68,234
,234,152,72,169,15,141,5,212,169
267 DATA255,141,6,212,169,48,141,1,212,1
69,17,141,4,212,32,102,64,169,0,141
268 DATA4,212,104,168,96,234,234,234,234
,234,32,122,65,32,68,65,32,103,65
269 DATA165,203,201,64,240,250,234,234,2
34,234,32,0,64,32,93,66,32,132,68
270 DATA240,25,32,160,70,32,132,68,240,1
7,32,17,68,162,20,160,0,234,136,208
271 DATA252,202,16,247,76,22,69,234,234,
206,227,7,169,176,205,227,7,240,18
272 DATA234,169,32,133,2,32,102,64,198,2
,208,249,32,128,67,76,6,69,234,234
273 DATA162,0,189,219,7,221,205,7,240,3,
16,23,96,234,232,224,5,208,239,234
274 DATA234,234,234,234,234,234,234,234,
234,234,234,96,234,162,0,189,219
275 DATA7,157,205,7,157,205,59,232,224,5
,208,242,96,165,252,197,254,144,3
276 DATA240,3,96,24,96,165,251,197,253,1
44,248,56,96,234,134,2,138,10,168
277 DATA185,0,208,133,20,169,0,133,21,56
,42,202,16,252,234,45,16,208,240
278 DATA4,169,1,133,21,234,56,165,20,233
,24,133,20,165,21,233,0,133,21,70
279 DATA21,102,20,70,21,102,20,70,21,102
,20,234,185,1,208,56,233,52,74,74
280 DATA74,170,224,0,240,16,24,165,20,10
5,40,133,20,165,21,105,0,133,21,202
281 DATA208,240,24,169,4,101,21,133,21,2
34,162,8,188,74,60,177,20,201,32
282 DATA240,22,201,81,240,18,201,71,240,
14,201,72,240,10,201,76,240,6,201
283 DATA77,240,2,56,96,202,16,222,24,96,
234,234,234,234,173,0,208,133,251
284 DATA173,16,208,41,1,133,252,234,234,
165,94,10,170,189,0,208,133,253,166
285 DATA94,169,0,56,42,202,16,252,234,45
,16,208,208,5,133,254,76,87,70,169
286 DATA1,133,254,234,32,140,69,208,1,96
,234,144,12,173,40,208,201,245,240
287 DATA13,166,94,76,123,70,234,173,40,2
08,201,245,240,243,166,94,76,142
288 DATA70,234,166,94,32,161,64,166,94,3
2,159,69,144,5,166,94,32,208,64,96
289 DATA234,166,94,32,208,64,166,94,32,1
59,69,144,5,166,94,32,161,64,96,234
290 DATA169,3,133,94,32,38,71,240,6,32,3
6,70,32,212,70,198,94,208,241,96
291 DATA234,234,138,10,170,189,1,208,24,
105,8,157,1,208,96,234,234,138,10
292 DATA170,56,189,1,208,233,8,157,1,208
,96,96,165,94,10,170,189,1,208,133
293 DATA253,173,1,208,133,251,169,0,133,
252,133,254,234,32,140,69,208,1,96
294 DATA144,26,173,40,208,201,245,240,27
,234,166,94,32,181,70,166,94,32,159
295 DATA69,144,5,166,94,32,196,70,96,173
,40,208,201,245,240,231,234,166,94
296 DATA32,196,70,166,94,32,159,69,144,5
,166,94,32,181,70,96,234,165,162
297 DATA170,189,0,160,41,3,201,0,96,234,
-1
300 :
310 :
320 :
330 INPUT"DATA=>MA. ODER TEST ";A
W$
340 IFAW$="D"THEN500
350 :
360 REM ******** TEST *********
370 :
380 READA:GE=GE+1:PF=PF+A:IFA<>-1THEN380
390 :
400 IFGE>5939THENPRINT"**** ZUVIELE DATE
N ****
410 IFGE<5939THENPRINT"**** ZUWENIG DATE
N ****
420 IFPF<>676734THENPRINT"**** PRUEFSUMM
E VERKEHRT ****
430 IFPF<>676734THENPRINT"DIFFERENZ:"676
734-PF
440 IFPF<>676734ORGE<>5939THENEND
450 :
460 :
470 REM ******** DATA-MA ********
480 :
490 RESTORE
500 PRINT"WENN DIE RICHTIGE DISKETTE EIN
LIEGT,    BITTE EINE TASTE DRUECKEN...
510 POKE198,0
520 GETA$:IFA$=""THEN520
530 :
540 OPEN1,8,2,"MA-PAC,PRG,WRITE"
550 PRINT#1,CHR$(0)CHR$(48);
560 FORT=1TO5938:READA:PRINT#1,CHR$(A);:
NEXT
570 PRINT#1,CHR$(13):CLOSE1
580 NEW
READY.
```

Listing »DATA-PAC« (Schluß)

# Von den Kleinen auf die Großen

**Für einen so erfolgreichen Computer wie den Commodore 64 dauert es naturgemäß nicht lange, bis auch entsprechend viel gute Software zur Verfügung steht. Viele von diesen Programmen, die nicht die speziellen Eigenschaften des C 64 benutzen, könnten ohne weiteres auf den CBM-Geräten der 3-, 4- und 8000-Serie laufen, wenn Commodore nicht überflüssigerweise den Beginn des Basic-RAM-Bereichs geändert hätte.**

Während jedoch der C 64 und auch der VC 20 in der Lage sind, auf CBM-Geräten erstellte Basicprogramme selbsttätig an das eigene System anzupassen (siehe dazu den entsprechenden Artikel in dieser Ausgabe), besitzen die CBM-Geräte diese Fähigkeit nicht. Es ist vielmehr Aufgabe des Anwenders die Programme anzupassen, will er etwa C 64-Programme auf den CBMs laufen lassen.

Im Heft 9/83 der Zeitschrift mc wurde eine einfache Möglichkeit vorgestellt, um auf C 64 erstellte Programme auch für die CBM-Geräte lauffähig zu machen. Es werden dabei mit Hilfe einer Zählschleife die Inhalte der Speicherstellen ab 2048 (Beginn des C 64-Basic-Bereichs) bis Programmende auf die Speicherplätze ab 1024 (Basic-Beginn für die CBMs) transferiert.

Diese Prozedur dauert je nach Länge des zu transferierenden Programms mitunter ein paar Minuten. Aus diesem Grund benutzte ich für die Umwandlung von C 64-Programmen ein eigens dafür erstelltes Maschinenprogramm. Ein versehentlich in den Maschinenprogramm-Bereich gePOKEter Wert sorgte eines Tages dafür, daß statt des ganzen Programms nur ein Block (256 Byte) transferiert wurde. Erstaunlich genug, das (ansonsten nicht lauffähige) Programm ließ sich anstandslos listen!

Diese zunächst überraschende Tatsache hat eine einfache Erklärung: Die Vorwärtszeiger (die ersten beiden Byte einer jeden Basic-Zeile) geben den Beginn der nächsten Basic-Zeile absolut an, das heißt es spielt keine Rolle, ob dieser weniger als 80 Byte oder ein ganzes KByte weiter liegt — die Zeile wird immer richtig angesprungen. Daß das Programm dann nicht auch noch lauffähig ist, liegt daran, daß der Basic-Interpreter, solange keine Sprunganweisungen vorkommen, sich nicht nach den Zeilenzeigern richtet, sondern die Zeilen byteweise in ihrer natürlichen Reihenfolge abarbeitet. Stößt er dabei auf eine Lücke zwischen den Zeilen, so kann er damit nichts anfangen, das System stürzt ab.

Folgende Befehlssequenz, etwa in Direktmodus eingegeben, liefert zum Beispiel die Anfangsadressen der Zeilen eines Basic-Programms:

```
x=1025: printx;: fori=1to1000: x=peek(x) + peek(x+1)
*256: printx;: ifx<>0thennext
```

Es wurde dabei vorausgesetzt, daß das vorliegende Programm nicht mehr als 1000 Zeilen Umfang hat. Will man die Arbeitsweise des Betriebssystems erkennen, so kann man in der obigen Zahlenfolge in eine der dort stehenden Adressen (jede Adresse: Zwei Byte in der Form, Adresse = LSB + MSB*256) statt der nächsten eine etwas später kommende Adresse einPOKEn. Alle dazwischen liegenden Zeilen verschwinden dann aus dem Listing, sie werden aber bei RUN vom Interpreter abgearbeitet. Die »verschwundenen« Zeilen werden wieder sichtbar, sobald eine neue Zeile mit RETURN »übernommen« wird.

Nehmen wir uns die Zeit, um uns an einem Beispiel klar zu machen, was eigentlich im letzten Absatz gemeint ist. Angenommen die Zeilen eines Basicprogramms ergeben die Zahlenfolge: 1025 1057 1092 1149 1226 1286 1350 1412 1465 1500

Das sind die Adressen, an denen jeweils eine Basic-Zeile beginnt, und zwar steht in den Speicherstellen 1025 und 1026 zusammen die Adresse 1057, in 1057 und 1058 die Adresse 1092 und so weiter. Wenn wir nun zum Beispiel in die Adresse 1092 (und 1093) etwa die Adresse 1350 einPOKEn könnten, so würden die Zeilen dazwischen vom Listing verschwinden, der Interpreter würde sich aber weiterhin durchlaufen und abarbeiten. Wie geht das POKEn vor sich? Wenn Ihr System den Befehl »doke« kennt, etwa durch eine Erweiterung wie Exbasic, so geben Sie einfach ein: doke1092,1350. Wenn nicht, so muß man sich erinnern, daß eine Adresse gleich dem Ausdruck MSB*256+LSB ist, wobei MSB und LSB für Most Significant (höherwertiges) Byte und Least Significant (niederwertiges) Byte steht. Es ist in unserem Beispiel: msb=int(1350/256): lsb=1350—msb*256.

msb und lsb sind hier richtige numerische Variablen und müssen (das gilt für die ersten zwei Zeichen, die allein relevant sind) ungeSHIFTet eingegeben werden! Sie geben jetzt ein: poke1092,lsb:poke1093,msb

Um ein C 64-Programm auf einem CBM listen zu können, ist es also nicht erforderlich, erst das ganze Programm zu verschieben, es genügt nur ein Teil davon. Genauer: ein Teil, der

```
1 print"■"tab(10)"■■■■■transfer C64 -> cbm■■■":print"aktivieren mit ■sys 960!■"
4 fori=0to22:readx:poke960+i,x:next:new:rem verschieb.c64->cbm
5 data162,0,189,1,8,157,1,4,232,208,247,165,43,56,233,4,133,43,133
6 data45,133,47,96


           Dr. Kaisar Petanides
```

**Listing des kurzen Programms für die Umsetzung von C 64-Programmen auf CBM-Computer**

mindestens eine Zeile enthält! Der Grund liegt darin, daß das System erst das Ende einer Zeile, gekennzeichnet durch eine 0, gefunden haben muß, bevor der Sprung in die nächste Zeile erfolgt.

Vorgehen: Man gibt im Direktmodus ein:
fori=1025to1111:pokei,peek(i+1024):next:poke43,peek(43)-4

Der Wert 1111 aus mnemonischen und Sicherheitsgründen (1111 - 1024 > 80, Anzahl der Zeichen in einer Basic-Zeile). Die »Verschiebung« dauert etwa eine Sekunde.

Man listet einige Zeilen des Programms, setzt den Cursor auf die erste Zeile und übernimmt sie mit RETURN. Dadurch werden auch alle anderen Zeiger richtig gesetzt, wobei das Betriebssystem gleichzeitig die einzelnen Zeilen lückenlos in steigenden Nummern aneinanderfügt. (Das ist übrigens auch der Grund, warum es zwar prinzipiell möglich, jedoch nicht einfach ist, etwa relokatible Maschinenprogramme zwischen den Basic-Zeilen zu verstecken. Über die Möglichkeit, Textteile in Basic-Zeilen zu verstecken, werden wir bei anderer Gelegenheit reden). Das Programm ist jetzt auch auf CBM-Geräten lauffähig, von systemgebundenen Befehlen natürlich abgesehen.

Aber auch diese Prozedur ist zeitraubend, wenn mehrere Programme von einem System auf das andere übertragen werden müssen. Das kleine Basic-Programm (siehe Listing) namens »transfer« wäre in diesem Fall eine bessere Alternative: Es erzeugt ein Maschinenprogramm ab Adresse 960 und löscht sich anschließend selbst. Das Maschinen»programm« bleibt, so lange der Computer nicht ausgeschaltet wird, in diesem Bereich resistent, verschiebt beim Abruf mit »sys 960« einen einzigen Block des C 64-Programms ab Adresse 2049 auf Adresse 1025 aufwärts und setzt auch die Zeiger für den Beginn der Variablen richtig.

Vorgehen: Das Basicprogramm »transfer« laden und mit RUN starten. Anschließend das zu verschiebende Programm laden, »sys 960« eintippen und mit RETURN abschließen. Dadurch werden die ersten 256 Byte des Basicprogramms »herunter« geholt. Das verschobene Programm läßt sich jetzt listen. Cursor auf die erste Zeile setzen und mit RETURN übernehmen. Gegebenenfalls eine eigene REM-Zeile als Zeile mit der niedrigsten Nummer übernehmen!

(K. Petanides/rg)

---

# User-Port-Display

**Mit dem User-Port-Display lassen sich viele Steuerungsprobleme lösen. Dabei ist es manchmal wichtig zu erkennen, ob die gewünschten Daten anliegen. Dieses Problem löst das folgende Programm.**

Nachdem ich mir im Sommer '83 einen C 64 zugelegt hatte, begann ich mich als vorheriger Nur-Elektroniker sofort für die Hardware und den User-Port des Computers zu interessieren. Während einiger Steuerexperimente kam mir die Idee zu diesem Programm.

Es eignet sich gut zum Austesten eigener Ein/Ausgabeoperationen. Möchte man zum Beispiel fremde Drucker anschließen, mit anderen Computern (zum Beispiel anderen C 64) kommunizieren oder irgendwelche Steuerungsprobleme lösen, so kann man mit Hilfe dieses Programmes jederzeit (auch während der Basic-Programmbearbeitung) erkennen, ob am User-Port die gewünschten Daten anliegen.

Die Routine ist in Maschinensprache geschrieben, das Listing zeigt den Basic-Lader. Nach dem Starten des Programms wird in der rechten oberen Ecke des Bildschirms nach einmaligem Aufruf ständig binär der momentane Zustand des Datenregisters angezeigt.

Mit SYS 49309 setzt das Programm den IRQ-Vector auf die eigentliche Anzeigenroutine.

Mit RUN/STOP-RESTORE wird der Zeiger wieder zurückgesetzt.

(Jan Legenhausen/gk)

```
10 rem ********************************
15 rem ***   userport-display      ***
20 rem ***                          ***
30 rem *** (c) by jan legenhausen ***
40 rem ***                          ***
42 rem ***        nocken 11         ***
44 rem ***                          ***
46 rem ***     56 wuppertal 11      ***
48 rem ***                          ***
50 rem ***      fuer den c-64       ***
55 rem ********************************
60 rem *** start mit sys (49309)  ***
70 rem ********************************
100 data 169, 48, 157, 30, 4, 169, 0, 15
7, 30, 216, 173, 1
110 data 221, 234, 96, 169, 192, 141, 1,
 192, 173, 1, 221, 162
120 data 0, 160, 46, 140, 0, 192, 41, 12
8, 208, 98, 32, 9
130 data 192, 232, 160, 59, 140, 0, 192,
 41, 64, 208, 85, 32
140 data 9, 192, 232, 160, 72, 140, 0, 1
92, 41, 32, 208, 72
150 data 32, 9, 192, 232, 160, 85, 140,
0, 192, 41, 16, 208
160 data 59, 32, 9, 192, 232, 160, 98, 1
40, 0, 192, 41, 8
170 data 208, 46, 32, 9, 192, 232, 160,
111, 140, 0, 192, 41
180 data 4, 208, 33, 32, 9, 192, 232, 16
0, 124, 140, 0, 192
190 data 41, 2, 208, 20, 32, 9, 192, 232
, 160, 137, 140, 0
200 data 192, 41, 1, 208, 7, 32, 9, 192
210 data 76, 49, 234, 234, 169, 49, 157,
 30, 4, 169, 0, 157, 30, 216, 173, 1
220 data 221, 108, 0, 192, 169, 24, 141,
 20, 3, 169, 192, 141
230 data 21, 3, 96
2000 fors=49161to49152+167:reada:zz=zz+a
:pokes,a:next
2100 ifzz<>17028thenprint"fehler in data
-zeilen !":stop
2110 sys49309
ready.
```

**Der Basic-Lader des Programms »User-Port-Display«**

# Autostart für den VC 20

**Will man Programme vor unbefugten Eingriffen schützen, so ist dies nur möglich, indem man die Programme auf EPROMs ab $A000 schreibt, die sich dann nach dem Einschalten des VC 20 selbst starten (Modulspiele, Toolkit). Es ist aber auch möglich, solche »Autostart«-Programme mit der Datasette und dem VC 20 zu generieren.**

Ein automatischer Programmstart ist überhaupt erst möglich, weil das Betriebssystem des Computers nach dem Laden eines Programmes vom Kassettenrecorder einen indirekten Sprung über einen Vektor der Zeropage zu einer Routine des Betriebssystems durchführt. Beim VC 20 findet dieser Sprung über den Vektor $0302/$0303 zum Basic-Warmstart des Interpreters statt. Dieser Vektor ist jetzt so zu verändern, daß er auf eine selbstgeschriebene Routine zeigt, die dann nach dem Laden eines Programmes von der Kassette sofort gestartet wird. Um nun eine Wirkung zu erzielen, muß dieser Vektor also während des Ladevorgangs an seine ursprüngliche Adresse ($0302/$0303) geschrieben werden. Normalerweise werden aber Programme, die mit SAVE gespeichert wurden, vom Computer als Basic-Programme angesehen und daher nicht unbedingt immer an ihre ursprüngliche Adresse geladen, sondern an den Anfang des Basic-Speichers verschoben. Speichert man dagegen die Programme mit SAVE "Name",1,1 ab, so werden sie später immer in den Speicherbereich geladen, in dem sie zum Zeitpunkt des SAVEns gestanden haben. Setzen wir doch zum Beispiel in einem Programm (Listing 1) den Basic-Warmstartvektor auf $FD22 (High Byte: 253/ Low Byte: 34) und die Basicanfang/ende-Zeiger auf den Bereich des LOAD-Vektors ($2B:01/$2C:03/$2D:04/$2E:03). Wenn wir diesen dann mit SAVE "RESET",1,1 abspeichern, erfolgt nach dem späteren Laden des Programms mit LOAD ein Sprung zu der Adresse, die durch den Vektor gekennzeichnet wird. In diesem Fall wird sofort nach dem Stoppen des Kassettenmotors ein Reset durchgeführt (SYS 64802).

## Ein kleiner Trick hilft weiter

Mit dieser Methode läßt sich aber nur der Basic-Warmstartvektor abspeichern, da das Programm im Basicspeicher steht und mit dem SAVE-Befehl nur zusammenhängende RAM-Bereiche gespeichert werden können. Daher wird ein Ladeprogramm benutzt, das das eigentliche Basicprogramm in den Programmspeicher lädt und selbst im unteren Bereich des Kassetten-Puffers ($033C-$03FC) steht. Dort wird es dann als Name (Tape-Header) zusammen mit dem veränderten Vektor auf die Kassette geschrieben. Damit ist auch der Wert des Vektors auf den Bereich des Kassetten-Puffers festgelegt.

Wird jetzt der Kassetten-Puffer mit dem Basic-Warmstartvektor (der auf den Anfang des Ladeprogramms, $0355, zeigt) und dem Befehl SAVE " ",1,1 auf eine Kassette geschrieben und wieder geladen, fängt der Computer nach dem Stoppen des Kassettenmotors sofort an, das Programm ab $0355 auszuführen. Statt des Ladeprogramms kann auch eine Hilfsroutine, die nicht länger als 192 Byte ist, in den Kassettenpuffer geschrieben werden, die dann nach dem Laden, ohne speziellen Startbefehl, sofort zur Verfügung steht. Schließlich wird das zu schützende Basicprogramm nach dem Vektor, ohne Tape-Header (Programmname), abgespeichert. Würde das Basic-Programm normal geSAVEt werden, bräuchte man das Autostart-Programm nur zu übergehen und könnte das zu schützende Basic-Programm auch so laden. Das Lade-Programm im Kassettenpuffer lädt dann das Basic-Programm in den Speicher. Danach werden die Vektoren $0318/0319 (NMI-Vektor) und $0308/$0309 (Text-Stop-Vektor) so geändert, daß erstens die Stop-Taste nicht mehr abgefragt wird und zweitens beim Versuch, das Basic-Programm mit RUN/STOP + RESTORE zu stoppen, das Basic-Programm abgebrochen wird und in einer Endlosschleife die Ausgabe eines Textes (»Autostart ist gesichert«) erfolgt. Zum Schluß wird das Basic-Programm mit RUN (in Maschinensprache: JSR $C659 +JMP $C7AE) gestartet.

## Bedienung des Programms »Autostart« (Listing 2)

1. Basic-Lader eintippen und noch einmal genau die DATA-Statements prüfen.
2. Basic-Lader mit RUN starten; daraufhin wird das Maschinenprogramm generiert und vom Basic-Lader aus mit SAVE "Autostart Gen.",1,1 gspeichert.
3. Laden des Maschinenprogramms "Autostart Gen." mit LOAD und Eingabe von: POKE 56,28:NEW.
4. Laden des zu schützenden Programms.
5. Autostart-Generator mit SYS 7168 starten.
6. Programmname auf Aufforderung hin eingeben; das zu schützende Programm wird dann als Autostart-Programm geSAVEt.

Es ist darauf zu achten, daß die Programme, die mit Autostart versehen werden, ausgearbeitete Programme sind, die nicht noch irgendeine Ergänzung oder Verbesserung benötigen. Da die Autostart-Programme nicht mehr zu stoppen sind, kann später auch nicht mehr an ihnen gearbeitet werden, so daß ein unfertiges Programm noch einmal neu eingetippt werden müßte.

```
10  POKE43,1:POKE44,3:POKE45,4:POKE46,3:CLR:
    REM Basicanfang/ende setzen
20  POKE770,34:POKE771,253:     :REM Basic-
    Warmstartvektor
30  SAVE "Reset",1,1
40  POKE770,131:POKE771,196:POKE43,1:POKE44,
    18:POKE45,185:POKE46,18:END
50  REM Zeiger und Vektoren wieder auf alten Wert
    bringen
```

Das Programm "Autostart" läuft nur auf einem VC 20 mit 3,5 KByte Speicher (Grundversion) sowie der Datasette und belegt die obersten beiden Pages von 7168 bis 7616. Die verschlüsselten Programme werden automatisch in den Grundversionsspeicher geschrieben. Daher ist es mit dieser Version nicht möglich, Programme, die auf einer Speichererweiterung laufen, mit "Autostart" zu versehen.

(Dirk Rother/ev)

```
0 POKE36879,8:PRINT" AUTOSTART":PRINT"
COPYRIGHT 1983":PRINT"BY D.ROTHER"
1 PRINT"PROGRAMM WIRD GELADEN (AB7168)
............."
2 POKE56,28:POKE55,0:CLR
3 FORI=7168TO7617:READA:POKEI,A:SU=SU+A:
NEXT:IFSU<>40847THENPRINT"TIPPFEHLER":EN
D
4 PRINT"BITTE  SPACE  DRUECKEN":WAIT1
98,1:PRINT"MASCHINENPROGRAMM WIRD GESPE
ICHERT"
5 POKE56,30:POKE44,28:POKE43,0:POKE46,29
:POKE45,200:SAVE"AUTOSTART GEN.",1,1
6 POKE43,1:POKE44,16:POKE45,172:POKE46,2
3:END
22 DATA169,8,141,15,144,169,3,141,134,2,
32,95,229,162,20,169,32,157,236,28,202,2
08
24 DATA250,169,168,160,28,32,30,203,162,
0,169,0,133,198,165,198,240,252,173,119
26 DATA2,201,13,240,13,157,236,28,41,63,
157,132,30,232,224,20,144,228,165,43,141
28 DATA131,29,165,44,141,132,29,165,45,1
41,133,29,165,46,141,134,29,173,2,3,72,1
73
30 DATA3,3,72,169,3,133,44,133,46,141,3,
3,169,1,133,43,169,4,133,45,169,85,141,2
32 DATA3,169,191,162,236,160,28,32,189,2
55,169,1,170,168,32,186,255,32,86,225,17
3
34 DATA131,29,133,43,133,193,173,132,29,
133,44,133,194,173,133,29,133,45,133,174
36 DATA173,134,29,133,46,133,175,104,141
,3,3,104,141,2,3,76,21,247,65,85,84,79,8
3
38 DATA84,65,82,84,13,32,40,67,41,49,57,
56,52,13,68,46,32,82,79,84,72,69,82,13,1
3
40 DATA80,82,79,71,82,65,77,77,78,65,77,
69,58,13,13,45,45,45,45,45,45,45,45,45,4
5
42 DATA45,45,45,45,45,45,45,45,45,45,13,
13,0,80,71,77,45,83,67,72,85,84,90,32,73
44 DATA32,32,32,32,32,32,32,32,32,91,228
,32,82,253,169,0,141,34,145,173,216,3,13
3
46 DATA193,133,43,173,217,3,133,194,133,
44,173,218,3,133,174,133,45,173,219,3,13
3
48 DATA175,133,46,32,61,246,169,3,141,25
,3,141,41,3,169,161,141,24,3,169,156,141
50 DATA40,3,169,255,141,34,145,32,89,198
,76,174,199,169,0,201,254,96,169,0,141,3
4
52 DATA145,32,95,229,169,8,141,15,144,16
9,1,141,134,2,169,189,160,3,32,30,203,76
54 DATA166,3,65,85,84,79,83,84,65,82,84,
13,32,32,32,73,83,84,13,71,69,83,73,67,7
2
56 DATA69,82,84,0,1,16,171,17,32,32,32,3
2,32,32,32,32,32,32,32,32,32,32,32,32,32
58 DATA32,32,32,32,32,32,32,32,32,32,32,
32,0,0,1,255,255,255,255,255,128,0,0,0,0
60 DATA255,255,255,255,32,32,32,32,32,32
,32,32,32,32,32,0,0
62 DATA2,3,72,173,3
READY.
```

**Listing 2. Basic-Lader für "Autostart"**

# View BAM

## Dieses Programm zeigt Ihnen sekundenschnell, wie und wo die Floppy Dateien und Programme abspeichert, welche Blöcke noch frei sind und welche nicht.

Das hier vorgestellte Programm stellt die sogenannte BAM der Disketten grafisch auf dem Bildschirm des C 64 dar. Das gesamte Programm ist in Maschinensprache geschrieben und daher sehr schnell. Es ist dem Viewbam-Programm, das von Commodore auf der Testdiskette mitgeliefert wird, in zweierlei Hinsicht überlegen:

1. ist es, wie bereits erwähnt, um ein Vielfaches schneller und
2. stellt es übersichtlich die gesamte Diskettenbelegung dar, wobei freie, belegte und nicht belegbare Blöcke gut zu erkennen sind.

Eine Hardcopy läßt sich auch leicht anfertigen (siehe Bild 1).

Das Programm braucht inklusive Ladezeit des Blockes 18/0, auf dem die BAM gespeichert ist, unter drei Sekunden zur Ausführung. (Im Vergleich zum Viewbam von Commodore, das 73 Sekunden benötigt.)

Es ist interessant zu sehen, wie das DOS Files abspeichert, welche Blöcke noch frei sind und welche nicht. Ich wurde durch die geringe Geschwindigkeit und die fehlende Möglichkeit des Viewbam von Commodore, alle Blöcke auf einmal zu betrachten, zu diesem Programm animiert.

Die Routine läuft auf dem C 64 von Commodore mit dem Floppy-Laufwerk VC 1541. Die Routine steht im geschützten Speicherbereich $C000-$C21C, dezimal 49152 bis 49693. Man kann sie auch in einem Basicprogramm aufrufen.

Mit diesem Programm wird in grafischer Darstellung ersichtlich, wieviele von den 664 Blöcken besetzt beziehungsweise frei sind.

Nach mühevoller Eingabe der Data-Zeilen und des Ladeprogramms sollte man dieses erst einmal auf Diskette sichern.

Eine Prüfsumme wird berechnet; dadurch vermeidet man eventuelle Eingabefehler.

Zuerst wird die BAM, die sogenannte »Block Availiability Map«, von der Floppy in den Speicher des Computers geladen. Die BAM ist im Block 18/0, das heißt Track 18, Sektor 0, auf jeder Diskette gespeichert.

**Die BAM umfaßt 256 Bytes, die im einzelnen folgende Bedeutung haben:**

| | |
|---|---|
| Bytes 0-1 | Track und Sektor vom ersten Directory-Block (meistens 18/1). |
| Byte 2 | Hier steht ein »A« (ASCII-Code 65) für das 4040 Format |
| Byte 3 | nicht belegt |
| Bytes 4-143 | Hier stehen die für dieses Programm wichtigen Informationen |
| Bytes 144-161 | Diskettenname, aufgefüllt mit geshifteten Leerzeichen |
| Bytes 162-163 | Disk ID |
| Byte 164 | Geshiftetes Leerzeichen |
| Bytes 165-166 | »2A«-ASCII Repräsentation für die DOS-Version |
| Bytes 166-167 | Geshiftete Leerzeichen |
| Bytes 168-255 | Nullen, unbenutzt |

Es existieren 35 Tracks, für jeden Track wurden in dem für uns wichtigen Bereich von Byte 4 bis Byte 143 vier Bytes reserviert. Die Tracks sind auch noch in Sektoren unterteilt, maximal 21 und mindestens 17, dies ist durch die Formel 2*Pi*Radius begründet, das heißt, daß bei kleinerem Radius nach innen weniger Raum zur Verfügung steht.

Das erste Byte eines jeden Viererblocks gibt die Anzahl der noch freien Blöcke in diesem Track an.

Die nächsten drei Bytes beinhalten die Informationen, welche Sektoren in dem Track belegt sind, und welche nicht.

Jedes Byte besteht aus 8 Bit und kann somit über 8 Sektoren Auskunft geben. Ein gesetztes Bit (1) bedeutet: freier, verfügbarer Block, ein nicht gesetztes Bit (0) bedeutet das Gegenteil, dieser Block (Track/Sektor) ist belegt.

Aus diesen Informationen kann man dann ein überschaubares Bild schaffen. (Jörg Schieb/gk)

```
0 DATA85,49,58,50,32,48,32,49,56,32,48,1
3,162,1,32,201,255,162,0,189,0
1 DATA192,32,210,255,232,224,12,208,245,
96,66,45,80,58,50,32,48,13,32,204
2 DATA255,162,1,32,201,255,162,0,189,31,
192,32,210,255,232,224,8,208,245
3 DATA96,169,1,162,8,160,15,32,186,255,1
69,0,32,189,255,32,192,255,169
4 DATA2,162,8,160,2,32,186,255,169,35,13
3,251,162,251,160,0,169,1,32,189
5 DATA255,32,192,255,32,12,192,32,39,192
,32,204,255,162,2,32,198,255,162
6 DATA0,32,207,255,157,0,142,232,208,247
,32,204,255,169,1,32,195,255,162
7 DATA0,189,194,193,32,210,255,232,224,5
8,208,245,162,144,189,0,142,32
8 DATA210,255,232,224,162,208,245,169,13
,32,210,255,162,0,189,252,193,32
9 DATA210,255,232,224,12,208,245,173,162
,142,32,210,255,173,163,142,32
10 DATA210,255,169,13,32,210,255,32,210,
255,169,5,32,210,255,160,20,152
11 DATA72,170,201,10,176,5,169,32,32,210
,255,169,0,32,205,189,169,45,32
12 DATA210,255,104,72,240,5,169,13,32,21
0,255,104,168,136,16,220,162,2,160
13 DATA3,32,10,229,160,4,169,48,162,9,20
8,10,192,1,240,4,162,10,208,2,162
14 DATA6,32,210,255,202,208,250,24,105,1
,136,208,234,162,3,160,3,32,10,229
15 DATA160,35,169,49,32,210,255,24,105,1
,201,58,144,2,169,48,136,208,241
16 DATA169,1,133,2,162,4,165,2,105,2,168
,32,10,229,162,20,134,139,169,142
17 DATA133,141,169,1,166,2,105,4,202,208
,251,133,140,165,139,166,2,224,18
18 DATA144,32,224,25,144,20,224,31,144,8
,201,17,144,20,162,8,208,42,201
19 DATA18,144,12,162,8,208,34,201,19,144
,4,162,8,208,26,165,139,74,74,74
20 DATA168,177,140,168,165,139,41,7,170,
152,61,8,194,208,4,162,0,240,2,162
21 DATA4,160,4,189,16,194,32,210,255,232
,136,208,246,165,139,240,5,169,17
22 DATA32,210,255,198,139,16,165,230,2,1
65,2,201,36,208,128,32,231,255,169
23 DATA0,133,198,165,198,240,252,96,147,
5,18,32,146,61,66,69,76,69,71,84
24 DATA44,46,61,70,82,69,73,44,78,61,48,
32,18,40,67,41,49,57,56,52,32,66
25 DATA89,32,74,46,32,83,67,72,73,69,66,
146,158,68,73,83,75,78,65,77,69
26 DATA32,58,32,157,157,157,157,157,157,
157,73,68,32,58,32,1,2,4,8,16,32
27 DATA64,128,18,32,146,157,158,46,5,157
,156,78,5,157,169,2
1000 REM ***************************
1010 REM * VIEWBAM FUER DEN VC64   *
1020 REM * (C) 1984 BY J. SCHIEB   *
1030 REM * RICHARD-WAGNER-STR.58   *
1040 REM * 4050 MOENCHENGLADBACH1  *
1050 REM *                         *
1060 REM * STARTEN MIT SYS 49213   *
1070 REM ***************************
1080 REM
1090 REM LADEROUTINE :
1100 FOR X=49152 TO 49693
1110 :   READ Y
1120 :   POKE X,Y
1130 :   S=S+Y
1140 NEXT
1150 IF S<>63349 THEN PRINT"** FEHLER IN
 DATAS !! **":END
1160 REM NEW

READY.
```

**Der Basic-Lader von »Vier BAM«**

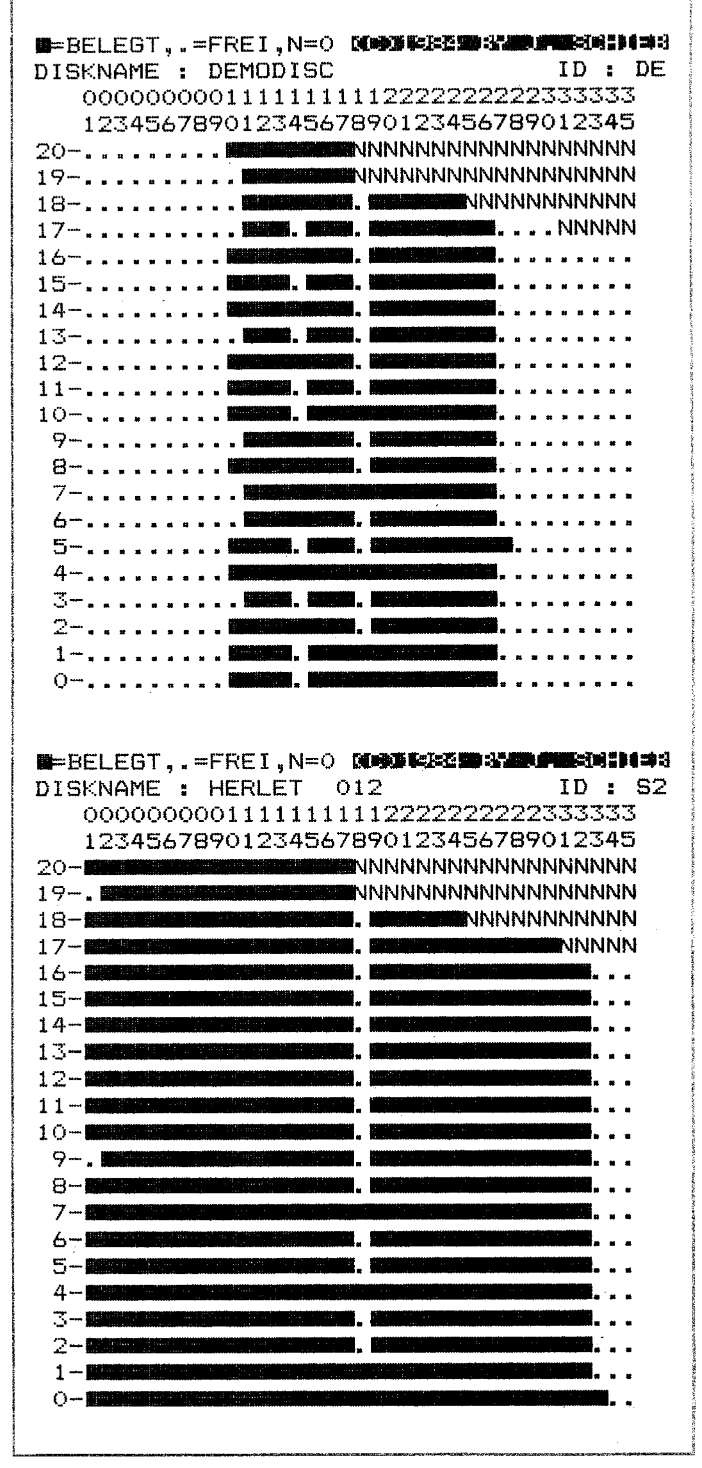



**Bild 1. Zwei Beispiele von »View BAM«**

# »PRINT AT« und »RESTORE N«

**Die hier vorgestellte kleine Basic-Erweiterung bringt zwei häufig benötigte Funktionen, nämlich die PRINT-Ausgabe an einer wählbaren Bildschirmposition und das Setzen des DATA-Zeigers auf einen bestimmten Datensatz. Zusätzlich ist noch die Möglichkeit gegeben, beliebige Bildschirmzeilen zu löschen.**

In vielen Leserbriefen wurde nach einem geeigneten »PRINT AT« gefragt. Da bislang keine optimale Lösung angeboten wurde, habe ich nun einen »PRINT AT-Ersatz« geschrieben. Diese Ersatzroutine arbeitet sehr zuverlässig und ist denkbar einfach anzuwenden. Das 146 Bytes lange Maschinenprogramm enthält außerdem eine Routine, mit der man beliebig viele Bildschirmzeilen löschen kann und eine Erweiterung des Restorebefehls, nämlich »RESTORE N«. Es ist nun möglich, die Data-Zeile, aus der das nächste Statement gelesen werden soll, selbst zu bestimmen, ohne die nichtbenötigten Daten zu überlesen. Der Datazeiger kann zum Beispiel mit »RESTORE 1100« direkt auf Zeile 1100 gestellt werden und beim ersten »READ« wird das erste Statement aus Zeile 1100 gelesen. Beim Umnumerieren eines Programmes mit diesem Befehl ist die Zeilenangabe von Hand zu ändern, da Renumberroutinen diesen Befehl nicht berücksichtigen. Der Restore-Befehl kann nach wie vor auch ohne Angabe einer Zeilennummer verwendet werden.

Für die Bildschirmzeilen-Löschroutine steht ein Kurzbefehl zur Verfügung. Die Syntax ist »£A,B«. In A steht die Nummer der obersten, in B die Nummer der untersten zu löschenden Zeile. Beispiele:

£0,22 löscht den ganzen Bildschirm
£0,0 löscht die oberste Zeile
£4,9 löscht BS-Zeilen 4 bis 9 (5. bis 10. von oben)

Das besondere an dieser Einrichtung ist, daß die Cursorposition dabei nicht verändert wird. Folgt der »£«-Befehl auf eine IF..THEN-Abfrage, so ist vor dem »£«-Befehl ein »:« zu setzen. Wird er vergessen, meldet der Interpreter einen »Syntax Error«. Die Werte von A und B müssen zwischen 0 und 22 liegen.

Auch die PRINT AT-Routine wird mit einem Kurzbefehl aufgerufen. Das Kurzzeichen ist der Klammeraffe (»@«) oder besser das »AT«-Zeichen. Die Syntax lautet: »@, X, Y; Ausdruck«. X steht für waagerecht, Y für senkrecht.

X darf Werte zwischen 0 und 21, Y Werte zwischen 0 und 22 erreichen. Andernfalls wird ein »Illegal Quantity Error« ausgegeben. Für X und Y sind möglich:

1. Direkte Zahlenangabe
2. Wertangabe in Form einer Variablen
3. Eine Formel als Koordinate.

Der Ausdruck kann Text (in Hochkommata eingeschlossen, Strichpunkt kann entfallen) oder eine Stringvariable (Strichpunkt unbedingt erforderlich) sein. Der Nullpunkt der Koordinaten ist die obere, linke Ecke. Einige Beispiele:

| | |
|---|---|
| @0,0"V C - 2 0" | Text beginnt oben links |
| @X,Y"V C - 2 0" | Text erscheint an gegebener Position |
| @A+B,3+A;A$ | String A$ wird gedruckt |
| @0,10; | Cursor wird für »Input« positioniert |

Auch hier ist bei einer IF...THEN Abfrage zwischen »THEN« und dem »@«-Befehl ein »:« zu setzen.

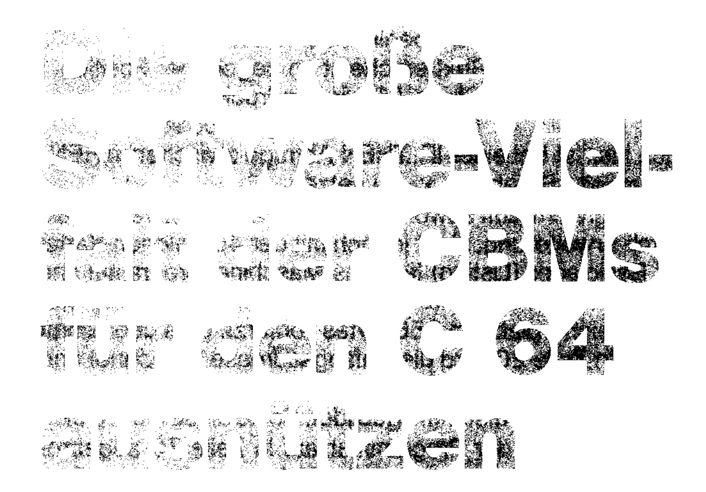
Das Programm läuft sowohl auf der VC 20-Grundversion als auch mit Erweiterung, nicht jedoch mit der 40-Zeichen-Karte.

Der Basiclader (Listing) lädt das Maschinenprogramm ans Speicherende. Hierzu wird der für Basic zur Verfügung stehende Speicherplatz um 256 Bytes verkürzt. Dies übernimmt Zeile 10. In Zeile 20 wird die Anfangsadresse berechnet und dann das Maschinenprogramm in den Speicher gepoket. Nebenbei wird eine Prüfsumme gebildet. Sollte diese nicht gleich 17442 sein, wird der Programmablauf in Zeile 70 abgebrochen. Ist der Prüfsummencheck OK (dies meldet das Programm), wird das Maschinenprogramm aufgerufen (Zeile 90). Abschließend meldet der VC 20 Interpreter »READ« und die Befehle stehen nun zur Verfügung. Wenn das Maschinenprogramm an einer bestimmten Stelle des Speichers stehen soll, kann dies durch Löschen der Programmzeile 10 und Ändern der Zeile 20 in »20 ES=[ADRESSE]« erreicht werden. Einen geeigneten Platz bietet zum Beispiel die 3KByte-Erweiterung, wenn Block 1 erweitert (8KByte oder 16KByte) ist. Auch der Kassettenpuffer (AB 828) ist verwendbar, jedoch nur bei Diskettenbetrieb empfehlenswert, da das Programm dann bei jeder Kassettenoperation gelöscht wird.

(Jürgen Reinert/ev)

```
1 REM ************************
2 REM * PRINT AT / RESTORE N *
3 REM *       (C) 1983       *
4 REM * BY JUERGEN REINERT   *
5 REM ************************
6 :
7 :
10 POKE56,PEEK(56)-1:CLR
20 ES=PEEK(55)+256*PEEK(56)+2
30 HB=INT((ES+11)/256)
40 LB=ES+11-HB*256
50 FORI=0TO145:READA:POKEI+ES,A
60 S=S+A:NEXT
70 IFS<>17442THENPRINT"▩DATA ERROR▩":END
80 PRINT"▩OK.▩"
90 POKEES+1,LB:POKEES+6,HB:SYSES:END
100 FORI=0TO145:READA:NEXT
110 DATA169,11,141,8,3,169,8
120 DATA141,9,3,96,169,199,72
130 DATA169,173,72,32,115,0,240
140 DATA244,201,140,240,8,201
150 DATA64,240,38,208,76,234
160 DATA234,32,115,0,240,26,201
170 DATA58,240,22,32,138,205
180 DATA32,247,215,32,19,198
190 DATA56,165,95,233,1,164
200 DATA96,176,1,136,76,39,200
210 DATA76,29,200,32,115,0,32
220 DATA158,215,224,22,176,24
230 DATA138,72,32,253,206,32
240 DATA158,215,104,224,23,176
250 DATA11,168,24,32,240,255
260 DATA32,121,0,76,160,202,76
270 DATA72,210,234,234,234,201
280 DATA92,240,4,56,76,237,199
290 DATA32,115,0,32,235,215,224
300 DATA23,176,232,164,20,192
310 DATA23,176,226,32,141,234
320 DATA202,228,20,16,248,76
330 DATA121,0,0,0,0
READY.
```

**PRINT AT und RESTORE N für den VC 20**




**Den meisten Anwendern wird sich schon einmal das Problem gestellt haben, wie sie Programme von anderen Computersystemen auf ihr eigenes übertragen können. Dieser Artikel sollte es jedem ermöglichen, auch komplexe Basicprogramme der weitverbreiteten Commodore-Systeme CBM 30XX, CBM 40XX (hier kurz als CBMs bezeichnet) und C 64 füreinander umzuschreiben.**

Hat man das Programm fertig eingegeben und auf Kassette oder Diskette abgespeichert, so ist das Laden von CBM-Programmen in den C 64 kein Problem: Disketten- und Kassettenformat sind identisch (zumindest bei den Doppellaufwerken 2031 und 4040 mit dem Einzellaufwerk VC 1541). Der Basic-Benutzerspeicher beginnt beim C 64 mit der Speicherstelle 2048 und endet bei 40959. Ein spezielles »Verschiebeprogramm« im C 64 sorgt dafür, daß CBM-Programme in den richtigen Speicherbereich geladen werden. Wer jedoch schon einmal versucht hat, C 64-Programme in einen CBM zu laden, wird damit keinen Erfolg gehabt haben, denn die CBMs besitzen kein solches Programm. Dort beginnt der Speicher für Basic-Programme bei 1024 und endet, je nach RAM-Ausbaustufe (8 KByte, 16 KByte oder 32 KByte) bei spätestens 32767. In einen CBM geladene C 64-Programme sind ohne Verbiegung von Zeigern weder zu listen noch zu starten. Man könnte bei den CBMs nach dem Laden den Zeiger für den Programmanfang verändern (durch den Befehl »POKE 40,0:POKE 41,8«), aber man müßte dies jedesmal nach dem Laden tun, ein auf die Dauer sehr umständliches Verfahren. Ein entsprechendes »Verschiebeprogramm«, welches diesen Aufwand vermeidet, finden Sie unter der Überschrift »Von den Kleinen auf die Großen« in dieser Ausgabe. Nun zur eigentlichen Arbeit, dem Umschreiben: Alle Computer haben die gleichen Basic-Anweisungen (bis auf die CBMs der Serie 4000, welche zusätzlich komfortable Diskettenbefehle besitzen). Sie unterscheiden sich im wesentlichen nur durch die Speicherbelegung und die Farbe, die Sprites und die hochauflösende Grafik des C 64.

**Der Bildschirmbereich:**

Die meisten Basicprogramme erstellen bewegte Grafiken mit Hilfe der Befehle PEEK und POKE. Im Bildschirmbereich wird durch den Befehl »POKE X, Y« an der Stelle mit der Adresse X das Zeichen mit dem Bildschirm-Code Y gesetzt. Gelöscht wird es dann durch den Befehl »POKE X, 32«, denn die Zahl 32 entspricht dem Leerzeichen (Blank). Durch PRINT PEEK(X) kann man abfragen, welches Zeichen sich an der Stelle X befindet. Der Bildschirmcode ist bei allen drei Systemen identisch. Jedoch liegt der Bildschirmspeicher an unterschiedlichen Stellen: Bei den CBMs zwischen 32768 und 33767, beim C 64 zwischen 1024 und 2023. Wenn man ein CBM-Programm auf den C 64 überträgt, muß man also von allen Bildschirmadressen 31744 subtrahieren, umgekehrt 31744 addieren.

Jedoch kommt da noch ein kleines Problem hinzu: Der C 64 kann 16 Farben darstellen. Dazu besitzt er einen Farbenspeicher, welcher ganz analog zum Bildschirmspeicher zu behandeln ist. Er beginnt bei 55296 und endet bei 56295; die Differenz zwischen Bildschirmadresse und Farbadresse beträgt 54272. Wenn man also ein CBM-Programm für den C 64 um-

| gedrückte Taste | CBM 30XX | CBM 40XX | Commodore 64 |
|---|---|---|---|
| ← | 75 | 95 | 57 |
| 1 | 26 | 49 | 56 |
| 2 | 18 | 50 | 59 |
| 3 | 25 | 51 | 8 |
| 4 | 42 | 52 | 11 |
| 5 | 34 | 53 | 16 |
| 6 | 41 | 54 | 19 |
| 7 | 58 | 55 | 24 |
| 8 | 50 | 56 | 27 |
| 9 | 57 | 57 | 32 |
| 0 | 10 | 48 | 35 |
| + | 17 | 43 | 40 |
| — | 9 | 45 | 43 |
| £ | -- | -- | 48 |
| HOME | 7 | 19 | 51 |
| INS/DEL | ? | ? | 0 |
| Q | 64 | 81 | 62 |
| W | 56 | 87 | 9 |
| E | 63 | 69 | 14 |
| R | 55 | 82 | 17 |
| T | 62 | 84 | 22 |
| Y | 54 | 89 | 25 |
| U | 23 | 86 | 30 |
| I | 53 | 73 | 33 |
| O | 60 | 79 | 38 |
| P | 52 | 80 | 41 |
| @ | 15 | 64 | 46 |
| * | 33 | 42 | 49 |
| ↑ | 59 | 94 | 54 |
| A | 48 | 65 | 10 |
| S | 40 | 83 | 13 |
| D | 47 | 68 | 18 |
| F | 39 | 70 | 21 |
| G | 46 | 71 | 26 |
| H | 38 | 72 | 29 |
| J | 45 | 74 | 34 |
| K | 37 | 75 | 37 |
| L | 44 | 76 | 42 |
| = | 1 | 61 | 53 |
| RETURN | 27 | 13 | 1 |
| Z | 32 | 90 | 12 |
| X | 24 | 88 | 23 |
| C | 31 | 67 | 20 |
| V | 23 | 86 | 31 |
| B | 30 | 66 | 28 |
| N | 22 | 78 | 39 |
| M | 29 | 77 | 36 |
| , | 70 | 44 | 47 |
| . | 2 | 46 | 44 |
| / | 49 | 47 | 55 |
| CRS↑ | ? | ? | 7 |
| CRS← | ? | ? | 2 |
| SPACE | 6 | 32 | 60 |

**Tabelle 1. Vergleich der Tastaturcodes**

schreibt, muß man hinter jedem POKE-Befehl für den Bildschirm einen entsprechenden für die Farbe anhängen, also zum Beispiel hinter POKE 1024,64 anfügen: POKE 1024 + 54272,X ,wobei X eine Zahl zwischen 0 und 15 ist, welche dann die Farbe des Zeichens bestimmt. Dieser Befehl kann weggelassen werden, wenn ein Zeichen gelöscht wird, also im Befehl der Bildschirmcode 32 (= Leerzeichen) eingesetzt wird. Die Farbcodes haben folgende Bedeutung:

| | |
|---|---|
| 0:schwarz | 8:orange |
| 1:weiß | 9:braun |
| 2:rot | 10:hellrot |
| 3:türkis | 11:grau 1 |
| 4:violett | 12:grau2 |
| 5:grün | 13:hellgrün |
| 6:blau | 14:hellblau |
| 7:gelb | 15:grau 3 |

Umgekehrt müssen die POKE-Befehle für den Farbenspeicher bei einer Übertragung auf die CBMs gestrichen werden. Sprites und hochauflösende Grafik können nicht auf die CBM-Serie übertragen werden.

## Tastatur und Joystick

Viele Programme fragen durch PEEK die gerade gedrückte Taste ab. Dazu existiert in der Zero-Page eine Speicherstelle, in welcher sich die Matrixkoordinate der gerade gedrückten Taste befindet. Bei den CBMs ist dies die 151, beim C 64 die 203. Die Wirkungsweise kann man sich leicht durch folgendes kleines Programm verdeutlichen:

```
10 PRINT PEEK (203)
20 GOTO 10
```

(Bei den CBMs muß die 203 in Zeile 10 durch 151 ersetzt werden). Nach dem Starten durch RUN erscheint eine Reihe von Zahlen, beim C 64 die 64, bei den CBMs die 255. Drückt man nun irgendeine Taste, so erscheint eine gerade Zahl, je nachdem, welche Taste gerade gedrückt ist. In Tabelle 1 ist ein Vergleich der Tastaturcodes dargestellt.

Es gilt, einige besondere Adressen zu beachten:
CBM: In 152 steht 1 (anstatt 0), wenn die Taste SHIFT gedrückt ist.
C 64: Adresse 653:
0: wenn nichts gedrückt
1: SHIFT gedrückt
2: Commodore-Taste gedrückt
4: CTRL gedrückt

Dazu noch ein Tip für eigene Programme: Meistens benötigt man für ein Spiel eine Unterscheidung zwischen verschiedenen Tastendrücken (zum Beispiel bis zu neun verschiedene Tasten für die Richtungssteuerung eines Raumschiffes). Am schnellsten funktioniert die Abfrage, wenn man ein Feld mit 256 (Ganzzahl-)Variablen definiert (zum Beispiel A%(255)), und die Richtung als Wert des entsprechenden Indexes darin speichert.

```
 1 REM KANONE
 5 ZA = 1984:ZE = 2023:OP = 2004
10 DIM A%(255)
20 A%(56) = —1:A%(59) = 1
:
1000 REM ABFRAGE — ZEICHEN SETZEN, SUBROUTINE
1010 NP = OP + A%(PEEK(203)):REM NEUE POSITION
1020 IF NP >ZE OR NP<ZA THEN NP = OP:REM ZEILENBEGRENZUNG
1030 POKE OP, 32:REM ALTE POSITION LÖSCHEN
1040 POKE NP, 30:REM NEUE POSITION SETZEN
1050 POKE NP+54272,1:REM FARBE SETZEN
1060 OP = NP
1070 RETURN
```

**Listing 1. Ein kleines Beispielprogramm für den C 64.**

Beispiel (hier für C 64): Wir wollen für ein Spielprogramm eine Kanone (↑ haben, die man am unteren Bildschirmrand mit Taste "1" nach links und mit Taste "2" nach rechts steuern kann. In unserem Beispiel würde das Programm entsprechend Listing 1 aussehen.
A% (255) ist das Variablenfeld. Der Tastaturcode von "1" ist beim C 64 die 56, von "2" die 59. Die Tastaturadresse ist 203.
Variablen:
OP die alte Position der Kanone
NP die neue Position der Kanone
ZE Bildschirmadresse am Ende der Zeile, in der sich die Kanone bewegt
ZA Bildschirmadresse am Zeilenanfang

CBM-Besitzer können nach Studium des Artikels an diesem Beispielprogramm ihr neugelerntes Wissen ausprobieren. Die Lösung steht am Ende des Artikels (aber nicht vorher nachschauen!).

Diese Methode ist auch bei der Verwendung eines Joysticks nützlich, denn dieser muß ähnlich wie die Tastatur mit PEEK abgefragt werden. Seine Adresse ist 56321 (Port 1) beziehungsweise 56320 (Port 2), wobei Port 1 weniger geeignet ist, da der Inhalt seines Speichers auch von der Tastatur beeinflußt wird. Die Codierung ist der Tabelle 2 zu entnehmen.

| Bewegung: | Port 1: | Port 2: |
|---|---|---|
| Feuer-Taste | 239 | 111 |
| nach oben | 254 | 126 |
| nach unten | 253 | 125 |
| nach rechts | 247 | 119 |
| nach links | 251 | 123 |
| nach links oben | 250 | 122 |
| nach rechts unten | 245 | 117 |
| nach rechts oben | 246 | 118 |
| nach links unten | 249 | 121 |
| keine Taste gedrückt | 255 | 127 |

**Tabelle 2. Die Codierung der Joystick-Ports**

Die CBMs besitzen keinen Anschluß für Joysticks, es gibt jedoch diverse Konstruktionen zum Anschluß an den User-Port. Daher empfiehlt es sich, beim Übertragen von C 64-Programmen auf die CBMs auf den Joystick zu verzichten und diese Programmteile durch Tastaturabfragen zu ersetzen.

## PRINT-Befehle

Die ASCII-Codes sind bei allen drei Systemen größtenteils identisch, sie unterscheiden sich praktisch nur in den Farbsteuerzeichen und der Umschaltung von Grafik auf Groß/Kleinschreibung. Die Umschaltung auf Groß/Kleinschreibung erfolgt bei den CBMs durch den Befehl "POKE 59468,14" und zurück auf die Blockgrafik durch "POKE 59468,12". C 64-Besitzer benutzen hier den Befehl "PRINT CHR$(14)" (Kleinschreibung) beziehungsweise den Befehl "PRINT CHR$(142)" (Grafik). Beim Übertragen von Programmen auf den C 64 kann man nach Belieben Farbsteuerzeichen einfügen, umgekehrt sind diese zu löschen.

## Der Ton

Wichtig für das Übertragen von Programmen mit Ton ist vor allem die Tonhöhe und ihre Adresse. Beim C 64 stehen drei Tongeneratoren zur Verfügung. Ihre Tonhöhe wird dabei durch je zwei Adressen bestimmt, wobei die erste Adresse (das Low-

Byte) die Tonhöhe nur sehr wenig, die zweite Adresse (das High-Byte) die Tonhöhe hingegen wesentlich stärker (nämlich genau 255 mal so stark wie das Low-Byte) beeinflußt. In der Tabelle 3 sind die wichtigsten Tonadressen des Commodore 64 zusammengefaßt.
(Adresse = 54272 + Register)

Bei den CBMs steht jedoch nur ein Tongenerator zur Verfügung, auch können dort weder Wellenform, noch Lautstärke, noch Hüllenkurve programmiert werden. Daher ist es bei der Übertragung von C 64-Programmen auf die CBMs bei komplexen Klängen und Geräuschen meist am einfachsten, den Ton selbst neu zu gestalten. Wenn man den Ton beim Übertragen auf den C 64 direkt übernehmen will, ist das Umschreiben nicht schwierig. Bei den CBMs gibt es drei Tonadressen, welche wie üblich durch PEEK und POKE angesprochen werden: 59464, 59466 und 59467. Der Tongenerator wird dort eingeschaltet durch POKE 59466,16:POKE 59467,20: POKE 59464,0. Nun kann man die Tonhöhe einfach festsetzen, indem man eine Zahl zwischen 0 und 255 in die Adresse 59646 schreibt (durch POKE 59464,X, wobei X eine Zahl zwischen 0 und 255 ist). Dabei ist 255 ein relativ tiefer Ton und 0 ein sehr hoher Ton (Ultraschall, daher unhörbar und mit »Ton ausgeschaltet« zu vergleichen). Die Tonhöhe kann man für den C 64 als High-Byte übernehmen, jedoch bedeutet dort 0 einen sehr tiefen Ton und 255 einen sehr hohen, also genau umgekehrt. Daher muß man wie folgt vorgehen. Wenn im CBM-Programm steht: POKE TA, TH, wobei TA die Adresse der Tonhöhe 59464 und TH die Tonhöhe ist, muß es nun im C 64-

| Stimme REGISTER 2 | 2 | 3 | INHALT | | |
|---|---|---|---|---|---|
| 0 | 7 | 14 | Frequenz Low-Byte (0-255) | | |
| 1 | 8 | 15 | Frequenz High-Byte (0-255) | | |
| 2 | 9 | 16 | Tastverhältnis Low-Byte (0-255) bei Rechteck | | |
| 3 | 10 | 17 | Tastverhältnis High-Byte (0-15) | | |
| 4 | 11 | 18 | Wellenform: | Rauschen = 129<br>Rechteck = 65<br>Sägezahn = 33<br>Dreieck = 17 | |
| 5 | 12 | 19 | Anschlag<br>0 (hart) —<br>15*16 (weich) | + | Abschwellen<br>0 (hart) —<br>15 (weich) |
| 6 | 13 | 20 | Halten<br>0 (stumm)—<br>15*16 (laut) | + | Ausklingen<br>0 (schnell)—<br>15 (langsam) |
| 24 | 24 | 24 | Lautstärke: 0 (stumm) — 15 (volle Lautstärke) | | |

**Tabelle 3. Die wichtigsten Tonadressen von C 64.**

| Doppelfloppy 4040<br>(die neuen 4er Befehle stehen in Klammern) | Monofloppy 1541 |
|---|---|
| Diskette neu formatieren: | |
| OPEN 1,8,15,"N0:Name,XX"<br>(HEADER"Name",IXX<br>I = Laufwerksnummer) | OPEN 1,8,15,"N:Name,XX" |
| Inhaltsverzeichnis lesen: | |
| LOAD"$1",8 bzw. LOAD"$",8<br>(DIRECTORY) | LOAD"$",8 |
| Beliebiges Programm laden: | |
| LOAD"Name",8<br>bzw. LOAD"Name1",8<br>(DLOAD"Name" bzw.<br>DLOAD"Name"D1) | LOAD"Name",8 |
| Beliebiges Programm sichern: | |
| SAVE"Name",8 bzw.<br>SAVE"Name1",8<br>(DSAVE"Name" bzw.<br>DSAVE"Name",D1) | SAVE"Name",8 |
| Datei löschen: | |
| OPEN 1,8,15,"S0:Name"<br>bzw. OPEN 1,8,15,"S1:Name"<br>(SCRATCH"Name",D0 bzw.<br>SCRATCH"Name",D1) | OPEN 1,8,15,"S:Name" |
| Datei umbenennen: | |
| OPEN 15,8,15:<br>PRINT15,"R0:Neu=0:Alt"<br>bzw.<br>OPEN 15,8,15:PRINT15,"R1:Neu=1:Alt"<br>(RENAME D1, "Alt"TO"Neu" bzw.<br>RENAME D0,"Alt"TO"Neu") | OPEN 15,8,15:<br>PRINT15,"R:Neu=Alt" |

**Tabelle 4. Vergleich der Befehle für die Laufwerke 4040 und 1541**

| CBM Hex | Dezimal | C 64 Hex | Dezimal | Bedeutung |
|---|---|---|---|---|
| 0028-0029 | 40-41 | 002B-002C | 43-44 | Zeiger auf Basic-Anfang |
| 002A-002B | 42-43 | 002D-002E | 45-46 | Zeiger auf Variablen Anfang |
| 002C-002D | 44-45 | s.u. | | Zeiger auf Ende der Variablentabelle |
| s.o. | | 002F-0030 | 47-48 | Zeiger auf Beginn der Felder |
| 002E-002F | 46-47 | 0031-0032 | 49-50 | Zeiger auf Ende der Felder |
| 0030-0031 | 48-49 | 0033-0034 | 51-52 | Zeiger auf Beginn der Strings (rückwärts) |
| 0032-0033 | 50-51 | 0037-0038 | 55-56 | Zeiger auf Ende der Strings (= Speichergrenze) |
| 0036-0037 | 54-55 | 0039-003A | 57-58 | laufende Zeilennummer |
| 0038-0039 | 56-57 | 003B-003C | 59-60 | vorhergehende Zeilennummer |
| 003A-0038 | 58-59 | 003D-003E | 61-62 | nächster Befehl (für CONT) |
| 003C-003D | 60-61 | 003F-0040 | 63-64 | aktuelle DATA-Zeile |
| 003E-003F | 62-63 | 0041-0042 | 65-66 | aktuelles DATA-Element (Adresse) |
| 0042-0043 | 66-67 | 0045-0046 | 69-70 | aktueller Variablenname |
| 0044-0045 | 68-69 | 0047-0048 | 71-72 | Zeiger auf aktuelle Variable |
| 0046-0047 | 70-71 | 0049-004A | 73-74 | Variablenzeiger für FOR...NEXT |
| 0070-0087 | 112-135 | 0073-008A | 115-138 | diese Routine holt nächstes Basic-Zeichen |
| 0088-008C | 136-140 | 008B-008F | 139-143 | Zufallszahl |
| 008D-008F | 141-143 | 00A0-00A2 | 160-162 | interne Uhr |
| 0097 | 151 | 00CB | 203 | gedrückte Taste |
| 00A7 | 167 | 00CC | 204 | Cursor an/aus (0=an,1=aus) |
| 00C6 | 198 | 00CA | 202 | Spaltenposition des Eingabecursors |
| 00D8 | 216 | 00C9 | 201 | Zeilenposition des Eingabecursors |
| 00FD-00FF | 253-255 | 00FB-00FE | 251-254 | freier Platz für Zeiger in Page 0 |
| 026F-0278 | 623-632 | 0277-0280 | 631-640 | Tastaturpuffer |

**Tabelle 5. Die wichtigsten Zero-Page-Adressen**

Programm heißen: POKE FH, 255 — TH. Es kommt aber dazu, daß der Tonumfang des C 64 etwas kleiner ist als der der CBMs. Daher empfiehlt es sich, den Wert für die Tonhöhe mit 2 oder 3 zu multiplizieren, wobei darauf zu achten ist, daß der Bereich von 0 bis 255 nicht überschritten wird. Der Befehl muß also lauten: POKE FH, 255 — 2 * TH.

## Diskettenbefehle

Die komfortablen Diskettenbefehle des CBM 40 XX können nicht übernommen werden, jedoch kann man die überall möglichen Kommando-String-Befehle verwenden. Allerdings muß dafür für die Monofloppy 1541 (beziehungsweise 1540) die Laufwerksnummer gestrichen werden. Tabelle 4 stellt eine Vergleichsliste der beiden Laufwerke dar.

Zum Schluß noch einige wichtige Zero-Page-Adressen zum Vergleich in Tabelle 5 (aus urheberrechtlichen Gründen dürfen die vollständigen Listen leider nicht abgedruckt werden).

Lösung des Beispielprogramms »Kanone« für CBM:

```
1 REM KANONE
5 ZA = 33728:ZE = 33767:OP = 33748
10 DIM A%(255)
20 A%(26) = —1:A%(18) = 1:REM (Für CBM 30XX)
oder
20 A%(49) = 1:A%(50) = 1:REM (Für CBM 40XX)
1000 REM Abfrage und Zeichen setzen, Subroutine
1010 NP = OP + A%(PEEK(151)):REM neue Position
1020 IF NP >ZE OR NP< ZA THEN NP = OP: REM Zeilenbegrenzung
1030 POKE OP, 32:REM alte Position löschen
1040 POKE NP,30:REM neue Position setzen
1050:
1060 OP = NP
1070 RETURN
```

Und nun viel Erfolg beim Umschreiben!

(M. und J. Heinz/rg)

# Tips & Tricks

## MERGE für C 64 / VC 20

Hier ist eine einfache MERGE-Routine zum Verbinden zweier Basic-Programme. Sie kann sowohl für die Floppy als auch für die Datasette (auch mit Turbo-Tape) verwendet werden. Einzige Voraussetzung: Das zweite Programm muß höhere Zeilennummern haben als das erste.

Und so wird's gemacht:

1. Sie laden das erste Programm. Dann geben Sie im Direktmodus ein: PRINT PEEK(43), PEEK(44) Diese beide Zahlen schreiben Sie sich auf.
2. Sie geben ein: POKE 43, (PEEK(45) + 256 * PEEK(46) - 2) AND 255 (Return) POKE 44, (PEEK(45) + 256 * PEEK(46) - 2) / 256 (Return)

Laden Sie nun das zweite Programm. Danach geben Sie ein: POKE 43, (erste Zahl) : POKE 44, (zweite Zahl) (Return) Nun befinden sich beide Programme hintereinander im Speicher.

(Michael Keukert)

## Funktionstastenbelegung

Simons Basic bietet ja bekanntlich die Möglichkeit, die Funktionstasten mit beliebigen Zeichenketten zu belegen. Um nun die Funktionstasten nicht jedesmal nach dem Einschalten neu belegen zu müssen, wäre es sinnvoll, die Belegung auf Floppy abspeichern zu können.

Die Funktionstastenbelegung ist bei Simons Basic in dem von Basic nicht erreichbaren Speicherbereich $C64D bis $C74B (50765 bis 51019 dezimal) abgelegt. Mit dem folgenden kleinen Programm wird dieser Speicherbereich als Maschinenprogramm abgespeichert:

```
10  INPUT »Filename»; X$
20  OPEN 5, 8, 5, X$ + »,P,W«
30  A = 50765 : E = 51019
40  H = INT(A / 256) : L = A AND 255
50  PRINT#5, CHR$(L); CHR$(H);
60  FOR I = A TO E
70  PRINT#5, CHR$(PEEK(I));
80  NEXT I : CLOSE 5
```

Mit LOAD "Name",8,1 kann die Funktionstastenbelegung nun jederzeit geladen werden, ohne ein eventuell vorhandenes Basicprogramm zu zerstören.

(Uwe Schwarz)

## Hilfe für »Turbo Tape«

Das Programm »Turbo Tape« ist ja ein Segen für alle diejenigen, die sich keine Floppy leisten können oder wollen. Es gibt allerdings einige Maschinenprogramme, die nach dem Gebrauch von »Turbo Tape« abstürzen.

Abhilfe: Nach dem Laden das Programm LISTen und den SYS-Befehl zu Anfang notieren. Nun SYS 64738 und danach den notierten SYS-Befehl eingeben — und schon läuft das Programm.

(Andreas Klofanda)

## Basicprogramme retten

Ein durch NEW oder durch einen RESET gelöschtes Basicprogramm kann durch Eingabe folgender Zeilen im Direktmodus wieder zurückgeholt werden:

POKE 46, PEEK(56) - 1 : POKE 45, PEEK(55) + 247 : CLR (Return)
POKE PEEK(44) * 256 + PEEK(43) + 1, PEEK(44) (Return)
63999 (Return)
FOR I = PEEK(44) * 256 + PEEK(43) TO PEEK(46) * 256 + PEEK(45) : IF PEEK(I) OR PEEK(I + 1) OR PEEK(I + 2) THEN NEXT (Return)
POKE 45, (I + 3) AND 255 : POKE 46, (I + 3) / 256 : CLR (Return)

Diese »Rettungsmaßnahme« funktioniert sowohl beim VC 20 wie auch beim C 64.

(Ralf Berle)

## Cursor steuern

Das Betriebssystem des C 64 enthält eine Routine, die man benutzen kann, um den Cursor an eine beliebige Stelle zu setzen. Geben Sie doch mal folgendes ein:

POKE 214, (Zeile) : POKE 211, (Spalte) : SYS 58640 : PRINT "TEXT"

(Michael Keukert)

## Und noch ein Tip

Der FORMULAR TOO COMPLEX - Error ist sehr unangenehm, da sich das Programm danach oft nicht mehr listen läßt. Nach Eingabe von POKE 24,0 verhält sich der Computer aber wieder normal.

(Roger Limberg)

# Listing des Monats: Castle of Doom

## Programm zu der Beschreibung von Seite 66

**Erklärung der Programmzeilen für Castle of Doom:**

Zur Erläuterung des Programms sind hier die verschiedenen Abschnitte des Programms aufgeführt:

| | |
|---|---|
| 1 | Sprung zur Befehlsausführung |
| 5—95 | DATAs einlesen, Erklärung? |
| 100—160 | Version auswählen |
| 165—176 | Zauberwörter bestimmen |
| 179—740 | Bilder zeichnen |
| 748—860 | Kommando auswerten |
| 870—940 | Bewegen |
| 950—980 | Sprung zur Befehlsausführung berechnen und ausführen |
| 999—5382 | Befehlsausführung |
| 6500—6528 | Befehl »sage« |
| 7000—7065 | Besitz + Gegenstände anzeigen |
| 7100—7125 | Befehl »nimm« |
| 7200—7415 | Befehl »wirf« |
| 7500—7560 | Erklärung |
| 7600—7720 | verloren |
| 7800—7880 | gewonnen |
| 8000—8110 | DATAs |

Da die Kommandoauswertung recht ungewöhnlich ausgeführt wird, folgt hier noch eine Beschreibung der betreffenden Programmzeilen:

| | |
|---|---|
| 755 | Kommandoeingabe |
| 760—785 | Zerlegen in Verb/Hauptwort |
| 795—810 | Vergleich mit der Liste aller bekannten Wörter<br>Zuordnung: X = Nummer des Verbs (1—14)<br>Y = Nummer des Hauptworts (1—21) |
| 830—845 | Untersuchen, ob eine sinnvolle Kombination von Verb und Hauptwort vorliegt |
| 960—980 | Aus X und Y wird die Zeilennummer berechnet, in der die Antwort auf das eingegebene Kommando steht. Die berechnete Zeilennummer wird in ihre Ziffern zerlegt, und diese werden in den Programmspeicher hinter das GOTO in Zeile 1 gePOKEd. Anschließend wird ein GOTO 1 ausgeführt. |

**Variablen-Definition zum Listing des Monats:**

Zum Abschluß hier noch die wichtigsten Variablen:

| | | | |
|---|---|---|---|
| V$(1...14) | Liste Verben | ZA = | 0,1,2 Zauberwort gesagt? |
| H$(1...21) | Liste Hauptwörter | K$ | Kommando |
| RI$(1...19) | mögliche Richtungen | V$ | Verb |
| ZI$(1...19) | Nachbarfelder | H$ | Hauptwort |
| G$(1...12) | Liste Gegenstände | KB$(1...9) | mögliche Kombinationen |
| G(1...12) | Ort der Gegenstände | MO = | 1 Monster besiegt |
| P | Position | GE = | 1 Geist besiegt |
| V | Version | MR = | 1 Monster anwesend |
| Z1$,Z2$ | Zauberwörter | GS = | 1 Geist anwesend |
| | | ZB = | 1 Zauberer anwesend |
| | | GG | Anzahl Gegenstände |

```
0 GOTO5
1 GOTO!!!!
2 REM
3 REM ZEILEN 0 UND 1 NICHT AENDERN !!!!
4 REM
5 DIMV$(14),H$(21),RI$(19),ZI$(19),G$(12
),G(12)
6 L$="                              "
7 CL$="|                          |"
8 EF$="
  "
10 POKE53280,14:POKE53281,14:POKE53272,2
3:P=15:QQ=54272:VI=53248
15 POKEVI+21,0:PRINT""SPC(12)"
-♠♥|L¯ Γ— ┬┬\":REM CASTLE OF DOOM
20 PRINT"     ¯IN ♠DVENTURE VON |.OEIS
SBECKER"
25 PRINTSPC(11)"* |ITTE WARTEN *"
30 FORI=1TO14:READV$(I)
35 IFI<10THENREADKB$(I)
40 NEXT
45 FORI=1TO21:READH$(I):NEXT
50 FORI=1TO19:READRI$(I),ZI$(I):NEXT
55 FORI=1TO12:G$(I)=H$(I):READG(I):NEXT
60 FORI=1TO4:READZX$(I),ZY$(I):NEXT
65 FORI=13TO15
70 FORJ=0TO62:READK:POKE64*I+J,K:NEXT:NE
XT
85 PRINT"  \OECHTEN ♥IE EINE ¯RKLAERUNG
 ? (\/╱)"
90 GETA$:IFA$="J"THEN7500
95 IFA$<>"N"THEN90
100 PRINT""SPC(10)"-♠♥|L¯ Γ— ┬┬\"
105 PRINT" |EI DIESEM ♠DVENTURE KOENN
EN ♥IE ZWI-"
110 PRINT" SCHEN DREI XERSIONEN WAEHLEN
 :":PRINT"  OOLLEN ♥IE :"
115 PRINT"  1 - ¯INE VERZAUBERTE ¯RINZE
SSIN":PRINTSPC(6)"RETTEN"
120 PRINT"  2 - ¯INEN BOESEN ♠AUBERER B
ESIEGEN"
125 PRINT"  3 - ¯INEN ♥CHATZ SUCHEN"
130 PRINT" |ITTE GEBEN ♥IE DIE ENTSPRE
CHENDE"
135 PRINT" ⌐ENNZIFFER EIN."
140 POKE198,0:WAIT198,1
145 GETI$:IFASC(I$)<49ORASC(I$)>51THEN14
0
150 V=VAL(I$):IFV<>2THEN160
155 RI$(13)="W-O":ZI$(13)="1214":RI$(9)=
"W-O-R":ZI$(9)="081018"
157 RI$(8)="N-S-O":ZI$(8)="051209"
160 IFV=1THENG(7)=1
165 FORI=1TO4:J(I)=1+4*RND(TI):NEXT
170 Z1$=ZX$(J(1))+ZY$(J(2))
175 Z2$=ZX$(J(3))+ZY$(J(4))
176 IFZ1$=Z2$THEN165
177 TI$="000000"
179 REM **** BILDER ZEICHNEN ****
180 PRINT"";:POKE53272,21:POKEVI+21,0:
POKEVI+23,0:POKEVI+29,0
185 FORI=1TO20:PRINTSPC(8)CL$:NEXT
190 FORI=1824TO1863:POKEI,99:POKEI+QQ,0:
NEXT
195 IFP>11THEN415
197 POKE53280,5:POKE53281,5
200 PRINT"";:FORI=1TO4:PRINTSPC(8+I)"\"
```

Listing des Monats: Castle of Doom

```
SPC(22-2*I)"/":NEXT
205 PRINTSPC(13)"□_______________"
210 FORI=1TO10:PRINTSPC(12)" |"SPC(14)"| "
:NEXT
215 PRINTSPC(13)"‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾□"
220 FORI=1TO5:PRINTSPC(13-I)"/"SPC(12+2*
I)"\":NEXT
225 FORI=1TOLEN(RI$(P))STEP2:A$=MID$(RI$
(P),I,1)
230 IFA$="N"THEN250
235 IFA$="W"THEN260
240 IFA$="O"THEN265
245 NEXT:GOTO270
250 FORJ=1402TO1562STEP40:FORK=0TO3:POKE
J+K,160:POKEJ+K+QQ,0:NEXT:NEXT
255 POKE1402,254:POKE1405,252:GOTO245
260 FORJ=1552TO1752STEP40:POKEJ,160:POKE
QQ+J,0:NEXT:POKE1752,105:GOTO245
265 FORJ=1575TO1775STEP40:POKEJ,160:POKE
QQ+J,0:NEXT:POKE1775,95:GOTO245
270 ONPGOTO275,740,300,315,740,325,345,7
40,355,375,395
275 ONVGOTO740,280,285
280 POKE2040,15:POKEVI+39,0:POKEVI,150:P
OKEVI+1,140:POKEVI+29,1:POKEVI+23,1
282 POKEVI+21,1:ZB=1:GOTO740
285 PRINT"▒▒▒▒▒▒▒▒▒▒"SPC(13)"_____":PRIN
TSPC(12)"/    /|"
290 PRINTSPC(11)"/    / | ":PRINTSPC(11)"┌
 | ":PRINTSPC(11)"| $$ | /"
295 PRINTSPC(11)"└____|/ SCHATZ":FORI=1TO2
000:NEXT:GOTO7830
300 IFLEFT$(RI$(3),1)="N"THEN740
305 FORJ=1402TO1562STEP40:FORK=0TO3
310 POKEJ+K,160:POKEJ+K+QQ,06:NEXT:NEXT:
PRINTSPC(16)"□□□□■SPIEGEL":GOTO740
315 PRINT"▒▒▒"SPC(13)"◥▒    ◤":PRINTSPC
(14)"◥▒  ◤":PRINTSPC(15)"| |"
320 FORI=1TO10:PRINTSPC(15)"┌┐":NEXT:GOT
O740
325 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒"SPC(13)"______":PR
INTSPC(13)"|_____|"
330 PRINTSPC(12)"/    /| ":PRINTSPC(11)"/
  //| ":PRINTSPC(10)"/    //‾"
335 PRINTSPC(9)"┌‾‾‾‾|//":PRINTSPC(9)"|
 |/":PRINTSPC(9)"└_____|  SCHLAFZIMMER"
340 GOTO740
345 IFGE=1THEN740
350 POKE2040,13:POKEVI+39,1:POKEVI,150:P
OKEVI+1,150:POKEVI+23,1:POKEVI+29,1
351 POKEVI+21,1:IFG(12)<>0THENFORI=1TO20
00:NEXT:GOTO7630
352 GS=1:GOTO740
355 PRINT"▒■▒"SPC(11)"BURGHOF / BRUNNEN"
:PRINTSPC(13)"▒┌┐┌┐┌┐┌┐┌┐┌┐┌┐┌┐"
360 PRINTSPC(17)"▒▒▒▒▒▒▒▒▒ ______":PRINTS
PC(16)"▒        "
365 PRINTSPC(16)"| ‾‾‾‾‾ |":PRINTSPC(16)"
|        |":PRINTSPC(16)"‾‾‾‾‾‾‾‾"
370 GOTO740
375 POKE214,14:PRINT
380 IFFA=1THENPRINTSPC(23)"▒◥▒    ◥":PRIN
TSPC(24)"◥▒    ◥":GOTO740
385 PRINTSPC(23)"_____":PRINTSPC(23)"\
\"
390 PRINTSPC(16)"QUADRAT \    \":PRINTSPC
(25)"‾‾‾‾‾":GOTO740
395 PRINT"▒▒▒▒▒▒▒▒▒▒"SPC(23)"_____":PRINT
SPC(22)" |\   \":PRINTSPC(22)" | ┌┬┐"
400 PRINTSPC(15)"└┘         | | |. |":PRINTSPC
(15)"└┘       | | | |"
405 PRINTSPC(14)"▒◤    ◣‾‾‾‾‾\|_|_|":PRINT
SPC(13)"▒◤      ◥"
410 PRINTSPC(21)"SCHRANK":GOTO740
415 IFP>14THEN445
417 POKE53280,8:POKE53281,8
420 PRINT"▒";:FORI=1TO6:PRINTSPC(8)"▒"L$
:NEXT
425 FORI=55504TO55514STEP2:POKEI,7:NEXT:
FORI=55517TO55527STEP2:POKEI,7:NEXT
430 FORI=1TO11:PRINT"▒"SPC(8)L$:NEXT:FOR
I=1TO3:PRINT"■"SPC(8)L$:NEXT
435 IFLEFT$(RI$(P),1)<>"N"THEN740
440 POKE214,11:PRINT"■":FORI=1TO5:PRINTS
PC(18)"▒     ":NEXT
442 GOTO740
445 POKE53280,14:POKE53281,14:ONP-14GOTO
555,450,485,500,540
450 PRINT"▒▒▒▒▒▒▒"SPC(20)" ▪":PRINTSPC(1
8)"▒    "
460 PRINTSPC(17)"■▟▒ /  ▪":PRINTSPC(16)"▒
◤ /    ▪"
465 PRINTSPC(15)" ▟▒ (        ▪":PRINTSPC(1
4)"▟◤   \        ▪"
470 PRINTSPC(13)"▒◤          \        ▪"
475 FORI=1TO7:PRINTSPC(8)L$:NEXT
480 PRINTSPC(9)"□□□□□□▒BERG"
481 IFLEN(RI$(16))=5THENPRINTSPC(9)"▒▒ST
EILER WEG NACH OBEN"
482 GOTO740
485 PRINT"▒"SPC(9)"▒▒▒▒STEILER BERGHANG,
":PRINTSPC(9)"▒HOEHLE"
490 PRINTSPC(21)"▒▒▒▁":PRINTSPC(19)" ▟▒
▪ ▀▄ "
495 FORI=1TO6:PRINTSPC(19)"▒          ":NEX
T:GOTO740
500 PRINT"▒";:FORI=1TO19:PRINTSPC(8)L$:N
EXT
505 PRINT"▒"SPC(9)"▒▒IM BRUNNEN ..."
510 IFV<>3THEN740
515 IFMO=1THENPRINT"▒":GOTO530
520 POKE2040,14:POKEVI+39,13:POKEVI,200:
POKEVI+1,150:POKEVI+23,1:POKEVI+29,1
525 PRINTSPC(9)"▒▒MONSTER":POKEVI+21,1:M
R=1
530 PRINTSPC(9)"▒▒TUER▒▒▒▒":FORI=1TO7:P
RINTSPC(16)"      ":NEXT
535 IFRI$(18)="H"THENPOKE1601,114
537 GOTO740
540 PRINT"▒";:FORI=1TO20:PRINTSPC(8)L$:N
EXT
545 PRINT"▒"SPC(9)"▒▒IN DER HOEHLE ..."
550 PRINTSPC(9)"▒▒STOCKDUNKEL HIER ...":
GOTO740
555 PRINT"▒▒▒"SPC(13)"┌‾‾‾‾‾‾‾‾‾‾‾┐":PRI
NTSPC(13)"| B A S A R |"
560 PRINTSPC(13)"└___________┘"
565 IFXX=1THEN655
570 PRINTSPC(9)"■▒EIN   BEFREUNDETER"
575 PRINTSPC(9)"▒HAENDLER IST BEREIT,":P
RINTSPC(9)"▒IHNEN ZWEI DER FOLGEN-"
580 PRINTSPC(9)"▒DEN GEGENSTAENDE ZU":PR
INTSPC(9)"▒UEBERLASSEN :"
585 PRINTSPC(9)"▒▒1■ GLASKUGEL ▒2■ KAEFI
G"
590 PRINTSPC(9)"▒▒3■ AMULETT   ▒4■ BUCH"
595 PRINT"▒▒▒▒  BITTE ENTSPRECHENDE NUMM
ERN EINGEBEN"
```

**Listing des Monats: Castle of Doom (Fortsetzung)**

```
600 GETI$:IFI$=""THEN600
605 IFASC(I$)<49ORASC(I$)>52THEN600
610 ONVAL(I$)GOSUB635,640,645,650
615 GETJ$:IFJ$=""ORJ$=I$THEN615
620 IFASC(J$)<49ORASC(J$)>52THEN615
625 ONVAL(J$)GOSUB635,640,645,650
630 XX=1:GG=GG+2:GOSUB7000:GOTO655
635 POKE1633,49:G(1)=0:RETURN
640 POKE1645,50:G(2)=0:RETURN
645 POKE1713,51:G(3)=0:RETURN
650 POKE1725,52:G(4)=0:RETURN
655 YY=0:FORI=1TO4:IFG(I)=0THENYY=YY+1
657 NEXT:IFYY<>2THEN710
660 PRINT"▩▩▩▩▩▩"SPC(9)"WOLLEN SIE DIE GEGEN-"
665 PRINTSPC(9)"▩STAENDE, DIE SIE HIER":PRINTSPC(9)"▩ERHALTEN HABEN, NOCH  "
670 PRINTSPC(9)"▩EINMAL UMTAUSCHEN ?":PRINTSPC(9)"▩        (J/N)"
675 PRINT:PRINTSPC(8)CL$:PRINT:PRINTSPC(8)CL$:GOSUB860
680 GETI$:IFI$="N"THEN710
685 IFI$<>"J"THEN680
690 FORI=1TO4:G(I)=20:NEXT:GG=GG-2
695 PRINT"▩▩▩▩▩":FORI=1TO9:PRINTSPC(8)CL$:NEXT
700 PRINT"▩▩▩▩▩▩▩"SPC(12)"WAEHLEN SIE !":PRINT"▩▩▩▩▩▩▩▩"
705 GOTO585
710 PRINT"▩▩▩▩▩▩":FORI=1TO9:PRINTSPC(8)CL$:NEXT
715 PRINT"▩▩▩▩":FORI=1TO5:PRINTSPC(13)"▌"SPC(11)"▐":NEXT
720 PRINTSPC(13)"◣┌───────────┐":PRINTSPC(13)"│┌─┐┌─┐┌─┐│"
725 PRINTSPC(13)"││▩●◣││▩●◣││▩●◣││"
730 PRINTSPC(13)"│└─┘└─┘└─┘│":PRINTSPC(13)"└───────────┘"
740 POKE214,18:PRINT:PRINTSPC(9)"■RICHTUNGEN : ▩"RI$(P):GOTO0750
748 REM **** KOMMANDOAUSWERTUNG ****
749 FORI=1TO2500:NEXT
750 ZA=0:GOSUB7000
755 GOSUB860:INPUT"▩ KOMMANDO :";K$
760 IFLEFT$(K$,1)=" "THENK$=RIGHT$(K$,LEN(K$)-1):GOTO760
765 IFLEN(K$)=1THEN870
770 FORI=1TOLEN(K$):IFMID$(K$,I,1)=" "THEN780
772 NEXT
775 GOSUB860:PRINT"             UNKORREKTE EINGABE !":GOTO0749
780 V$=LEFT$(K$,I-1):H$=RIGHT$(K$,LEN(K$)-I)
785 IFLEFT$(H$,1)=" "THENH$=RIGHT$(H$,LEN(H$)-1):GOTO785
790 IFV$="SAGE"THEN6500
795 FORI=1TO14:IFLEFT$(V$(I),LEN(V$))=V$THENX=I:GOTO0805
800 NEXT:NV$=V$:GOTO0815
805 FORI=1TO21:IFLEFT$(H$(I),LEN(H$))=H$THENY=I:GOTO0825
810 NEXT:NV$=H$
815 GOSUB860:PRINT"□ TUT MIR LEID, ABER ICH VERSTEHE"
820 PRINT"▩ '"NV$"' NICHT.":GOTO0749
825 IFX>9THENX=X-5
830 FORI=1TOLEN(KB$(X)):IFMID$(KB$(X),I,1)=CHR$(Y+64)THEN950
835 NEXT:GOSUB860
840 PRINT"□ TUT MIR LEID, ABER ICH VERSTEHE IHREN"
845 PRINT"▩ BEFEHL NICHT.":GOTO0749
860 POKE214,20:PRINT:FORI=1TO3:PRINTEF$:NEXT:PRINT"□□□":RETURN
869 REM **** BEWEGEN ****
870 FORI=1TO6:IFK$=MID$("NSWOHR",I,1)THEN880
875 NEXT:GOTO0775
880 IFGS=1ORZB=1ORMR=1THEN7600
882 FORI=1TOLEN(RI$(P))STEP2
885 IFMID$(RI$(P),I,1)=K$THEN900
890 NEXT:GOSUB860
895 PRINT"   IN DIESE RICHTUNG GEHT ES NICHT":GOTO0749
900 SC=0:ZI=VAL(MID$(ZI$(P),I,2))
905 IFP=16ANDZI=13THEN925
910 IFP=13ANDZI=9ANDG(8)=0ANDV=1THEN7660
915 IFZI=0THEN7680
920 P=ZI:GOTO0180
925 IFG(8)=0THENONVGOTO0930,7650,7670
930 GOSUB860:PRINT"□ HINTER IHNEN LOEST SICH EINE STEIN-"
935 PRINT"▩ LAWINE, DIE DEN WEG VERSCHUETTET.":RI$(16)="W-O":ZI$(16)="1517"
940 FORI=1TO2500:NEXT:P=ZI:GOTO0180
949 REM ** SPRUNG ZUR BEF.AUSFUEHRUNG **
950 IFX=4ANDY<13THEN7100
955 IFX=5THEN7200
960 ZN=1000+(X-1)*500+(Y-1)*20:ZN$=STR$(ZN)
965 FORI=1TO4:NZ=VAL(MID$(ZN$,I+1,1))
970 POKE(2060+I),(NZ+48)
975 NEXT
980 GOSUB860:GOTO01
998 REM *** BEFEHLSAUSFUEHRUNG ***
999 PRINT"  OK !":GOTO0749
1000 IFY=1ORY=7ORY=16ORY=19ORY=20THENE$="E ":GOTO01002
1001 E$=" "
1002 PRINT" HIER IST KEIN"E$H$(Y):GOTO0749
1020 IFG(Y)=0ORG(Y)=PTHEN999
1022 GOTO01000
1060 GOTO01020
1180 IFG(10)<>0ANDG(10)<>PTHEN1000
1182 IFV<>2THENPRINT" DER "H$(10)" LAESST SICH NICHT OEFFNEN":GOTO0749
1184 IFG(9)<>21THEN999
1186 PRINT" IM "H$(10)" BEFINDET SICH EIN "H$(9)" !!":G(9)=P:GOTO0749
1300 IFP=18ANDRI$(18)="H"ANDV=3THEN1306
1302 IFP=1ORP>13THEN1000
1304 PRINT" HIER IST KEINE GESCHLOSSENE TUER":GOTO0749
1306 IFG(11)<>0THENPRINT" DIE TUER IST VERSCHLOSSEN !":GOTO0749
1308 RI$(18)="O-H":ZI$(18)="1909":RI$(19)="S-W":ZI$(19)="1718":GOTO0180
1340 IFP<>11THEN1000
1342 IFV=3ORG(10)<>21THEN999
1344 IFG(11)<>0THENPRINT" DER SCHRANK IST VERSCHLOSSEN":GOTO0749
1346 PRINT"□   IM SCHRANK BEFINDET SICH"
1348 PRINT"▩   EIN GOLDENER BALL":G(10)=11:GOTO0749
1400 PRINT" DAS IST UNMOEGLICH !":GOTO0749
1580 IFG(5)<>PTHEN1000
1582 IFP=18THENPRINT" ES IST ZU DUNKEL H
```

```
IER !":GOTO749
1584 IFG(2)=0THEN1590
1586 PRINT"□ SIE BRAUCHEN EINEN BEHAELTE
R"
1588 PRINT"▩ UM DEN FROSCH ZU FANGEN !":
GOTO749
1590 IFGG=5THENPRINT" SIE TRAGEN SCHON 5
 GEGENSTAENDE !":GOTO749
1592 G(5)=0:GG=GG+1:PRINT" OK !":GOTO749
1740 IFMR=1THEN7640
1742 IFV=3ANDP=9ANDMO=0THENPRINT" VON HI
ER AUS GEHT DAS NICHT !":GOTO749
1744 GOTO1000
1780 IFGS<>1THEN1000
1782 PRINT"  DER GEIST MAG DAS NICHT !!"
:FORI=1TO2000:NEXT:GOTO7630
2080 IFG(5)=PTHENPRINT" DER FROSCH LAESS
T SICH NICHT KUESSEN":GOTO749
2082 IFG(5)<>0THEN1000
2084 IFV<>1ORP<>6THEN999
2086 PRINT" ES KNALLT UND STINKT...":FOR
I=1TO2000:NEXT:GOTO7800
2240 IFMR<>1THEN1000
2242 PRINT"□ DAS MONSTER SCHLAEGT WILD U
M SICH"
2244 PRINT"▩ UND SCHLEUDERT SIE AUS DEM
BRUNNEN":MR=0
2246 FORI=1TO3000:NEXT:P=9:GOTO180
2260 IFZB<>1THEN1000
2262 FORI=0TO15:POKE53281,I:POKE53280,I:
FORJ=1TO100:NEXT:NEXT:ZB=0
2264 POKE53280,5:POKE53281,5:ZK=ZK+1:IFZ
K=3THEN7620
2266 P=2+INT(13*RND(TI)):GOTO180
2280 IFGS<>1THEN1000
2282 IFTI$<"001000"THEN7630
2284 GE=1:GS=0:FORI=832TO892STEP3:POKEI,
0:POKEI+1,0:POKEI+2,0:NEXT
2286 PRINT"□  DER GEIST HAT IHREN MUNDGE
RUCH":PRINT"▩  NICHT VERTRAGEN !"
2288 GOTO749
2400 IFP<>10ORFA=1THEN1000
2402 IFV<>1THEN999
2404 IFV=1THENFA=1:RI$(10)="N-W-R":ZI$(1
0)="070919"
2406 RI$(19)="S-H":ZI$(19)="1710":GOTO18
0
2820 IFP<>3ORLEN(RI$(3))=5THEN1000
2822 PRINT" DAS GEHT NICHT !":GOTO749
2840 IFP<>11THEN1000
2842 PRINT" DER SCHRANK IST ZU SCHWER !"
:GOTO749
2900 IFP<>10ORFA=1THEN1000
2902 GOTO2822
3560 IFGS=1THEN7630
3562 IFG(4)<>0THENPRINT" ICH HABE DAS BU
CH NICHT !":GOTO749
3564 IFV=3THEN3568
3566 PRINT" ZAUBERWORT : "Z1$:GOTO749
3568 PRINT"□ ZAUBERWOERTER : "Z1$:PRINT"
▩"SPC(17)Z2$:GOTO749
3660 IFG(9)<>0THENPRINT" ICH HABE KEINEN
 ZETTEL":GOTO749
3662 PRINT" ZAUBERWORT: "Z2$:FORI=1TO250
0:NEXT
3664 PRINT"□ DER ZETTEL VERSCHWINDET !!!
":GG=GG-1:G(9)=21:GOTO749
3860 IFP<>12ORSC=0THEN1000
3862 PRINT"□  TUT MIR LEID, ABER ES SCHE
INT"
3864 PRINT"▩  EINE FREMDE SPRACHE ZU SEI
N":GOTO749
4300 IFP<12OR(V=3ANDP=18)THENPRINT" OK..
.NICHTS PASSIERT !":GOTO749
4302 GOTO1000
4320 IFP<>3ORLEN(RI$(3))=5THEN1000
4322 PRINT"□ HINTER DEM "H$(17)
4324 PRINT"▩ SCHEINT EIN HOHLRAUM ZU SEI
N !":GOTO749
4380 IFV=2ANDP=12ANDLEN(RI$(12))=3THEN43
88
4382 IFP=19ANDG(8)=21THEN4392
4384 IFP>14ANDP<18THEN1000
4386 PRINT" OK...NICHTS BESONDERES FESTZ
USTELLEN !":GOTO749
4388 POKE214,11:PRINT:PRINTSPC(18)"▩▩,+/
0":GOSUB860
4390 PRINT" EINE SCHRIFT ERSCHEINT...":S
C=1:GOTO749
4392 G(8)=19:PRINT" AN DER WAND HAENGT E
IN SCHWERT !":GOTO749
4400 IFP=10ANDFA=0THENPRINT" KLINGT HOHL
...":GOTO749
4402 GOTO1000
4580 IFG(5)<>0ANDG(5)<>PTHEN1000
4582 PRINT" DER FROSCH ENTWISCHT IHNEN !
"
4584 IFG(5)<>18ORV<>1THEN4588
4586 GOTO749
4588 IFG(5)=0THENGG=GG-1
4590 G(5)=1+INT(19*RND(1)):GOTO749
4740 IFMR=0THEN1000
4742 PRINT" GUTE IDEE, ABER DAS GEHT NIC
HT !":GOTO749
4760 IFZB=0THEN1000
4762 GOSUB4770:IFY=0ORG(Y)<>0THEN749
4764 IFY=8ANDZA=2THEN7820
4766 GOTO7620
4770 INPUT" WOMIT ";WM$
4771 FORI=1TOLEN(WM$):IFMID$(WM$,I,1)="
"THEN4773
4772 NEXT:GOTO4774
4773 WM$=RIGHT$(WM$,LEN(WM$)-I)
4774 FORY=1TO12:IFWM$=LEFT$(H$(Y),LEN(WM
$))THEN4776
4775 NEXT:GOSUB860:PRINT" ICH VERSTEHE
"WM$"' NICHT !":Y=0:RETURN
4776 IFG(Y)<>0THENGOSUB860:PRINT" ICH HA
BE DIESEN GEGENSTAND NICHT !"
4777 RETURN
4780 IFGS=0THEN1000
4782 GOSUB4770:IFY=0ORG(Y)<>0THEN749
4784 GOSUB860:ONVGOTO4785,4790,4794
4785 IFG(8)=0THENPRINT" SIE SIND GESTOLP
ERT, UND...":FORI=1TO2000:NEXT:GOTO7630
4786 IFZA=1ANDY=7THEN4798
4788 GOTO7630
4790 IFZA=1ANDY=3THEN4798
4792 GOTO7630
4794 IFY=1THENPRINT" DER GEIST IST TOT,
ABER...":FORI=1TO2000:NEXT
4796 GOTO7630
4798 POKEVI+21,0:GS=0:GE=1:PRINT" OK...D
ER GEIST IST TOT !":GOTO749
5000 IFG(1)<>0ANDG(1)<>PTHEN1000
5002 PRINT" WOZU SOLLTE DAS GUT SEIN ?":
GOTO749
5020 IFG(2)<>0ANDG(2)<>PTHEN1000
```

**Listing des Monats: Castle of Doom (Fortsetzung)**

```
5022 PRINT"□ SO GEHT MAN NICHT MIT FREMD
EM"
5024 PRINT"▩ EIGENTUM UM !":GOTO749
5300 IFP=18ANDRI$(18)="H"ANDV=3THEN5306
5302 IFP=1ORP>13THEN1000
5304 PRINT"WIESO, HIER IST KEINE VERSCHL
OSSENE TUER":GOTO749
5306 PRINT" DIE TUER IST ZU STABIL...":G
OTO749
5320 GOSUB4770:IFY=0ORG(Y)<>0THEN749
5322 IF(V=1ANDY=8)OR(V=3ANDZA=2ANDY=8)TH
EN7247
5324 GOSUB860:PRINT" DAS GEHT NICHT !":G
OTO749
5340 IFP<>11THEN1000
5342 PRINT"□ DER SCHRANK LAESST SICH NIC
HT"
5344 PRINT"▩ MIT GEWALT OEFFNEN !":GOTO7
49
5380 IFP>14ANDP<18THEN1000
5382 PRINT"SIE KOENNEN DIE WAND NICHT
RSTOEREN !":GOTO749
6499 REM *** BEFEHL 'SAGE' ***
6500 GOSUB860:IFH$=Z1$THENZA=1:GOTO6506
6502 IFH$=Z2$THENZA=2:GOTO6506
6504 GOTO7690
6506 ONVGOTO6508,6512,6520
6508 IFZA=1ANDG(7)=0ANDGS=1ANDG(4)=0THEN
PRINT" DIE KEULE VIBRIERT...":GOTO6528
6510 GOTO6526
6512 IFZA=1ANDP=12ANDSC=1THENRI$(12)="N-
W-O":ZI$(12)="080013":GOTO415
6514 IFZA=1ANDG(3)=0ANDGS=1THENPOKEVI+39
,0:GOTO755
6516 IFZA=2ANDG(8)=0ANDZB=1ANDG(4)=0THEN
PRINT" DAS SCHWERT LEUCHTET...":GOTO6528
6518 GOTO6526
6520 IFZA=1ANDG(3)=0ANDMO=0ANDP=9THENPRI
NT" DAS AMULETT LEUCHTET...":GOTO6528
6522 IFZA<>2ORG(4)<>0ORP<>3ORRI$(3)<>"W-
O"THEN6526
6524 PRINT" DER SPIEGEL LEUCHTET...":GOT
O6528
6526 PRINT" OK...ES TUT SICH NICHTS !":G
OTO749
6528 FORI=1TO2000:NEXT:GOTO755
6999 REM * BESITZ/GEGENSTD ANZEIGEN *
7000 PRINT"▩";:FORI=1TO20:PRINT"
":NEXT
7002 PRINT"▩■▩SIE":PRINT"▩BESITZEN":SP=0
7005 FORI=1TO12:IFG(I)=0THENGOSUB7050
7010 NEXT
7020 PRINT"▩";:FORI=1TO20:PRINTSPC(32)"
      ":NEXT
7022 J=0:PRINT"▩■▩"SPC(32)"ES GIBT":PRIN
TSPC(32)"▩HIER :":SP=32
7025 FORI=1TO12:IFG(I)=PTHENGOSUB7050:J=
J+1:IFJ=6THENRETURN
7030 NEXT:RETURN
7050 IFI=1THENPRINTSPC(SP)"▩GLAS-":PRINT
SPC(SP)"KUGEL":RETURN
7055 IFI=11THENPRINTSPC(SP)"▩SCHLUES":PR
INTSPC(SP)"SEL":RETURN
7060 IFI=12THENPRINTSPC(SP)"▩KETTEN-":PR
INTSPC(SP)"HEMD":RETURN
7065 PRINTSPC(SP)"▩"G$(I):RETURN
7099 REM *** BEFEHL 'NIMM' ***
7100 IFG(Y)=PTHEN7110
7105 GOSUB860:PRINT"  DIESER GEGENSTAND
IST NICHT HIER !":GOTO749
7110 IFGG<5THEN7120
7115 GOSUB860:PRINT"  SIE HABEN BEREITS
5 GEGENSTAENDE !":GOTO749
7120 IFY=5THENGOSUB860:PRINT"    DER FRO
SCH HUEPFT IHNEN DAVON !":GOTO749
7125 G(Y)=0:GG=GG+1:GOSUB860:PRINT"   OK
 !":FORI=1TO1000:NEXT:GOTO750
7199 REM *** BEFEHL 'WIRF' ***
7200 IFG(Y)<>0THENGOSUB860:PRINT" ICH HA
BE DIESEN GEGENSTAND NICHT !":GOTO749
7205 GOSUB860:INPUT"  WOHIN ";WH$
7210 FORI=1TOLEN(WH$):IFMID$(WH$,I,1)="
"THEN7220
7215 NEXT:GOTO7225
7220 WH$=RIGHT$(WH$,LEN(WH$)-I)
7225 IFWH$="BODEN"ORWH$="WEG"ORWH$="EGAL
"ORWH$="WAND"THEN7235
7230 ONVGOTO7240,7270,7330
7235 GOSUB860:PRINT"  OK !":G(Y)=P:GG=GG
-1:GOTO749
7240 IFY<>8ORWH$<>"SPIEGEL"ORP<>3ORLEN(R
I$(3))=5THEN7255
7245 G(Y)=3:GG=GG-1
7247 RI$(3)="N-W-O":ZI$(3)="010204":GOSU
B860
7250 PRINT"  DER SPIEGEL ZERBRICHT !":FO
RI=1TO2000:NEXT:GOTO180
7255 IFY<>10ORWH$<>"BRUNNEN"ORP<>9ORG(5)
<>18THEN7380
7260 GOSUB860:PRINT"□  AUS DEM BRUNNEN K
OMMT EIN FROSCH"
7265 PRINT"▩  MIT DEM BALL IM MUND.":G(1
0)=9:G(5)=9:GG=GG-1:GOTO749
7270 IFY<>3ORWH$<>"GEIST"ORGS<>1THEN7290
7275 IFZA<>1THEN7630
7280 POKEVI+21,0:GOSUB860:PRINT"    SIE
HABEN DEN GEIST BESIEGT !"
7285 GE=1:GS=0:G(Y)=7:GG=GG-1:GOTO749
7290 IFY<>6ORP<>10ORWH$<>"QUADRAT"ORFA=1
THEN7305
7295 FA=1:G(6)=10:GG=GG-1:RI$(10)="N-W-R
":ZI$(10)="070919"
7300 RI$(19)="S-H":ZI$(19)="1710":GOTO18
0
7305 IFY<>4ORP<>3ORWH$<>"SPIEGEL"ORLEN(R
I$(3))=5THEN7315
7310 GOTO7245
7315 IFY<>8ORZB<>1ORWH$<>"ZAUBERER"THEN7
380
7320 IFZA<>2THEN7620
7325 POKEVI+21,0:FORI=1TO2000:NEXT:GOTO7
820
7330 IFY<>1ORWH$<>"GEIST"ORGS<>1THEN7340
7335 GOTO7280
7340 IFY<>3ORWH$<>"BRUNNEN"ORP<>9THEN736
0
7345 G(3)=18:GG=GG-1:GOSUB860:IFMO=1THEN
PRINT"  OK !":GOTO749
7350 IFZA=1THENPRINT" AUS DEM BRUNNEN ER
TOENT EIN SCHREI...":MO=1:GOTO749
7355 G(3)=9:PRINT" DAS AMULETT KOMMT ZUR
UECKGEFLOGEN !":GOTO749
7360 IFY<>8ORWH$<>"SPIEGEL"ORP<>3ORLEN(R
I$(3))=5THEN7380
7365 IFZA=2THEN7245
7370 GOTO7235
7380 IFWH$="BRUNNEN"ANDP=9THENG(Y)=18:GO
TO7415
7382 IFWH$="QUADRAT"ANDP=10ANDFA=0THENG(
Y)=10:GOTO7415
7385 IFWH$="HOEHLE"ANDP=17THENG(Y)=19:GO
TO7415
```

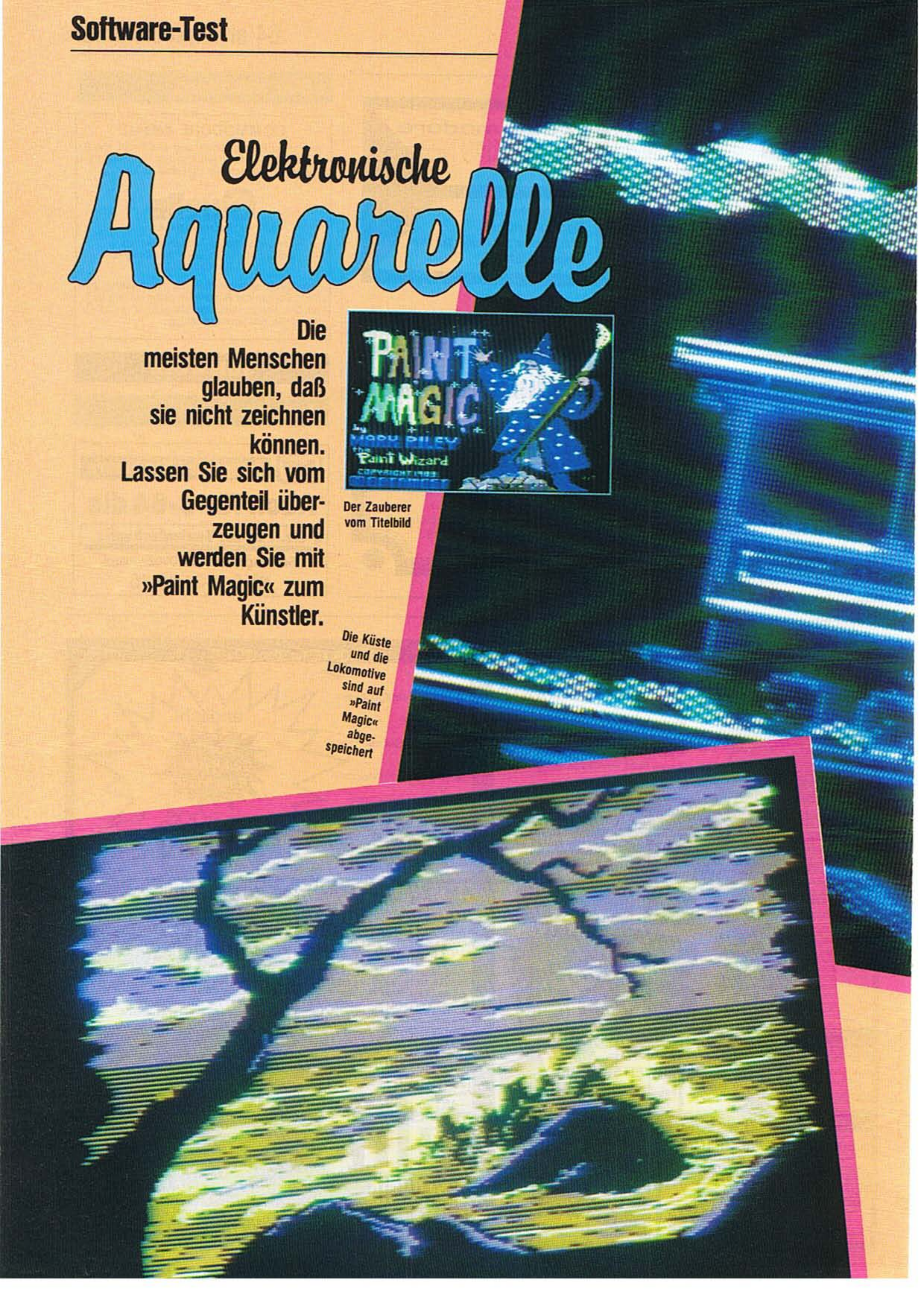
```
7390 IFWH$="GEIST"ANDGS=1THEN7630
7395 IFWH$="ZAUBERER"ANDZB=1THEN7620
7400 IFWH$="SPIEGEL"ANDP=3ANDLEN(RI$(3))
=3THEN7235
7405 GOSUB860:PRINT" ICH VERSTEHE SIE NI
CHT..."
7410 FORI=1TO2000:NEXT:GOTO7205
7415 GG=GG-1:GOSUB860:PRINT"  OK !":GOTO
749
7499 REM *** ERKLAERUNG ***
7500 PRINT"▩▩ ♥IE MUESSEN VERSUCHEN, EIN
E ♣UFGABE ZU"
7505 PRINT"▩ LOESEN, INDEM ♥IE DEM ─OMPU
TER SAGEN,"
7510 PRINT"▩ WAS ER TUN SOLL."
7515 PRINT"▩ ♦UR ─ORTBEWEGUNG  GEBEN  ♥
IE NUR"
7520 PRINT"▩ ♣BKUERZUNGEN EIN (/─♥─o─┌─|
─_ = /┌_─ ─":REM N-S-W-O-H-R = NORD -
7525 PRINT"▩ ♥,─── ─ o─♥|  ─ ┌♥|  ─  |┌─| ─
─,/| ─_)"
7526 REM SUED - WEST - OST - HOCH - RUNT
ER
7530 PRINT"▩ ♣NSONSTEN MUESSEN DIE |EFEH
LE AUS ZWEI"
7535 PRINT"▩ oOERTERN BESTEHEN (XERB/ |AU
PTWORT)"
7540 PRINT"▩ ─ER ─OMPUTER VERSTEHT DIE X
ERBEN :"
7545 PRINT"▩ OEFFNE,FANGE,KUESSE,NIMM,LI
ES,WIRF,"
7550 PRINT"▩ BERUEHRE/KLOPFE,ERSCHLAGE/T
OETE,SAGE"
7555 PRINT"▩ ZERBRECHE/ZERSTOERE    -   _E
RTIG ???"
7560 POKE198,0:WAIT198,1:GOTO100
7599 REM **** VERLOREN ****
7600 IFZB=1THEN7620
7605 IFGS=1THEN7630
7610 IFMR=1THEN7640
7620 PRINT"▩▩ DER ZAUBERER HAT SIE IN E
INE KROETE"
7625 PRINT"▩ VERWANDELT.":GOTO7700
7630 PRINT"▩▩      DER GEIST HAT SIE GEB
ISSEN.":GOTO7700
7640 PRINT"▩▩ DAS MONSTER HAT BESCHLOSS
EN, SIE"
7645 PRINT"▩ ZU HEIRATEN.":GOTO7700
7650 PRINT"▩▩ SIE SIND UNTERWEGS GESTOL
PERT UND"
7655 PRINT"▩ HABEN SICH DABEI DEN KLEINE
N FINGER":PRINT"▩ VERSTAUCHT.":GOTO7700
7660 PRINT"▩▩ AUS DEM TORBOGEN HAT SICH
 EIN STEIN"
7665 PRINT"▩ GELOEST UND IST IHNEN AUF D
EN FUSS":PRINT"▩ GEFALLEN.":GOTO7700
7670 PRINT"▩▩ SIE SIND IN EINEN ERDRUTS
CH GERATEN"
7675 PRINT"▩ UND HABEN SICH IHRE NEUE HO
SE DRECKIG":PRINT"▩ GEMACHT.":GOTO7700
7680 PRINT"▩▩ SIE SIND VOM BERG GEFALLE
N UND HABEN"
7685 PRINT"▩ DABEI IHRE BRILLE ZERBROCHE
N.":GOTO7700
7690 PRINT"▩▩         SIE HABEN SICH IN
EINE"
7695 PRINT"▩          BANANE VERWANDELT
!"
7700 PRINT"▩ DESHALB MUESSEN SIE LEIDER
AUFGEBEN."
7705 PRINT"▩▩▩▩ MOECHTEN SIE ES NOCH EIN
MAL PROBIEREN ?"
7710 GETI$:IFI$="J"THENRUN5
7715 IFI$<>"N"THEN7710
7720 END
7799 REM **** GEWONNEN ****
7800 PRINT"▩▩▩          ... UND  DER  "H$(
5)" ...":FORI=1TO1500:NEXT
7805 PRINT"□            ... VERWANDELT  SIC
H ...":FORI=1TO1500:NEXT
7810 PRINT"□... IN EINE WUNDERSCHOENE PR
INZESSIN ...":FORI=1TO2000:NEXT
7815 PRINT"▩▩▩▩▩▩":GOTO7840
7820 POKEVI+21,0:PRINT"▩▩▩▩▩▩▩▩    SIE H
ABEN DEN ZAUBERER BESIEGT !!!":GOTO7840
7830 PRINT"▩▩▩▩▩▩▩▩    SIE HABEN DEN SCH
ATZ GEFUNDEN !!!"
7840 PRINTSPC(13)"▩▩GRATULIERE,":PRINT"▩
     SIE HABEN IHR ZIEL ERREICHT !!"
7850 PRINT"▩▩▩        WOLLEN SIE WEITER SP
IELEN ?"
7860 GETI$:IFI$="J"THENRUN5
7870 IFI$<>"N"THEN7860
7880 END
8000 DATAOEFFNE,BDJPRU,FANGE,EMO,KUESSE,
EMNOU,NIMM,ABCDEFGHIJKLQRU
8005 DATAWIRF,ABCDEFGHIJKL,LIES,DIS,BERU
EHRE,PQTU,TOETE,EMNO
8010 DATAZERSTOERE,ABPQRT,WERFE,LESE,KLO
PFE,ERSCHLAGE,ZERBRECHE
8015 DATAGLASKUGEL,KAEFIG,AMULETT,BUCH,F
ROSCH,HELM,KEULE,SCHWERT,ZETTEL,BALL
8020 DATASCHLUESSEL,KETTENHEMD,MONSTER,Z
AUBERER,GEIST,TUER,SPIEGEL,SCHRANK
8025 DATASCHRIFT,WAND,QUADRAT
8030 DATAS,03,S-O,0503,W-O,0204,W-H,0311
,N-S,0208,O,07,S-W,1006,N-O,0509
8035 DATAS-W-O-R,13081018,N-W,0709,R,04,
W-O,0013,N-W-O,091214,W-O,1300,O,16
8040 DATAW-O-H,151713,N-W,1916,H,09,S,17
8045 DATA20,20,20,20,18,2,21,21,21,21,6,
5
8050 DATASILBER,BLUME,GOLD,RING,KUPFER,H
ORN,PLATIN,STERN
8054 REM SPRITE 1 - GEIST
8055 DATA0,127,0,0,255,192,1,206,96,3,25
5,240,1,255,240,1,245,248,1,255,248
8060 DATA3,255,248,7,255,248,15,255,252,
31,255,254,3,255,240,3,255,224
8065 DATA3,255,224,7,255,224,7,255,224,7
,255,240,15,255,240,15,255,248
8070 DATA31,255,252,59,187,110
8074 REM SPRITE 2 - MONSTER
8075 DATA12,0,48,30,0,120,3,0,192,1,129,
128,3,195,192,15,231,240,62,126,124
8080 DATA124,60,62,248,60,31,254,126,127
,255,255,255,127,255,254,127,255,254
8085 DATA63,90,252,30,165,120,31,255,248
,7,255,224,4,126,32,4,60,32
8090 DATA31,0,248,42,129,84
8094 REM SPRITE 3 - ZAUBERER
8095 DATA0,15,224,0,63,128,0,127,128,0,2
19,0,1,255,0,3,255,128,31,255,240
8100 DATA127,255,252,127,255,252,111,255
,236,111,255,236,103,255,204,98,170,140
8105 DATA99,255,140,71,255,196,7,199,192
,15,131,224,15,1,224,14,0,224
8110 DATA28,0,112,60,0,120

READY.
```

**Listing des Monats: Castle of Doom (Schluß)**

# Elektronische Aquarelle

**Die meisten Menschen glauben, daß sie nicht zeichnen können. Lassen Sie sich vom Gegenteil überzeugen und werden Sie mit »Paint Magic« zum Künstler.**




Der Zauberer vom Titelbild

Die Küste und die Lokomotive sind auf »Paint Magic« abgespeichert



Mit diesem Programm wird Ihr Computer zwar nicht zum Zauberer und es hat auch nichts mit Magie zu tun, wenn man mit »Paint Magic« arbeitet. Es bietet aber doch einige außergewöhnliche Fähigkeiten, die es von anderen Zeichenprogrammen abhebt. Sein Leistungsumfang resultiert allein daraus, daß es alle Hardwaremerkmale des C 64 nutzt.

Anhand einer leicht verständlich geschriebenen Anleitung (es lag leider nur eine englische Version vor) arbeitet man sich recht schnell in die Grundbefehle von »Paint Magic« ein. Mit diesen Befehlen kann man Linien ziehen, Boxen beziehungsweise Rechtecke und Kreise ziehen oder einfach frei auf dem Bildschirm zeichnen. Doch nur mit die-

# Elektronische Aquarelle



Schon der Anfänger schafft Kunstwerke

sen Möglichkeiten könnte man noch nicht viel anfangen. Beschäftigt man sich nun intensiver mit dem Befehlssatz und der Bedeutung der einzelnen Begriffe, lernt man bald die Vielfalt der Verwendungsmöglichkeiten und die Stärken von »Paint Magic« kennen.

## Auf zwei Bildschirmen gleichzeitig malen

Grundsätzlich hat man zwei Bildschirme zur Verfügung, zwischen denen man hin und her schalten kann. Setzt man den RESTORE-Befehl, bei dem der zuletzt getätigte Vorgang rückgängig gemacht wird, richtig ein, kann man sogar mit drei Bildern gleichzeitig arbeiten. Ein breites Spektrum an Verwendungsmöglichkeiten bietet der FILL-Befehl. So kann man nicht nur Flächen mit einer Farbe ausfüllen, sondern in Verbindung von zwei Farben eine horizontale, vertikale oder diagonale Schraffur erzeugen. Doch damit noch nicht genug. Hat man sich Farbmuster selbst definiert, kann man auch mit diesen den FILL-Befehl anwenden.

## Den Pinsel selbst gestalten

Eine interessante und neue Sache ist der GRAB-Modus. In diesem kann man sich ein Rechteck, daß maximal 40 x 40 Punkte beträgt, aus dem Bildschirm »herausgreifen« und dieses sozusagen als Pinsel verwenden. Außerdem arbeiten auch die Befehle LINE, BOX und CIRCLE im GRAB-Modus. Es ist damit sehr leicht, eindrucksvolle Grafiken auf den Bildschirm zu »zaubern«.

Sie werden sich schon fragen: Gibt es denn keine Möglichkeit, einen Ausschnitt zu vergrößern und damit zu zeichnen? Auch daran wurde gedacht. Drückt man die SPACE-Taste, kommt man in den MAGNIFY-Modus und der Bereich um den Cursor wird stark vergrößert dargestellt. Man kann dann mit dem Ausschnitt über das ganze Bild wandern und Veränderungen vornehmen. Bei all den oben beschriebenen Modi und Befehlen kann die Geschwindigkeit des Cursors, der mit einem Joystick oder über die Cursortasten bewegt wird, in acht Stufen variiert werden.

## Probleme bei den Farben

Die Einstellung der verschiedenen Farben ist so eine Sache bei »Paint Magic«. Man hat zwar ein eigenes Farbmenü, in dem man die fünf Arbeitsfarben vorwählen und vier Farbmuster aus diesen entwerfen kann. Doch man weiß nie genau, mit welcher Farbe man gerade arbeitet, wenn das Menü nicht mehr zu sehen ist. Die einzelnen Farben werden durch Drücken von SHIFT und eins bis fünf eingestellt. Die Arbeitsfarbe, die dann benutzt wird, ergibt sich durch Drücken von eins bis fünf alleine. Erst nach längerem Umgang mit »Paint Magic« hat man dieses Problem einigermaßen im Griff.

Das Programm verzichtet ganz auf den Einsatz von Ton, obwohl er an einigen Stellen recht nützlich wäre. Bei der Bedienung der Diskette findet man nur Befehle zum Laden, Abspeichern und Löschen von Bildern sowie zur Anzeige des Inhaltsverzeichnisses der Diskette. Es wäre wünschenswert, auch Diskettenbefehle einsetzen zu können, die alle Möglichkeiten zur Floppydiskbedienung enthalten.

Mit »Paint Magic« bekommt man ein sehr leistungsfähiges Werkzeug zur Ausschöpfung der Grafikfähigkeiten des Computers. In Verbindung mit dem gut geschriebenen und übersichtlich aufgebauten Handbuch stellt es eine ideale Lösung für solche Leute dar, die gerne auf dem Bildschirm malen, zeichnen oder einfach ihrer Fantasie freien Lauf lassen wollen.

## Hilfestellung

Wer sich nicht traut, gleich frei zu zeichnen, kann eines der elf abgespeicherten Bilder auf den Bildschirm holen und dieses verändern oder einfach darauf weitermalen. Zu diesem Vorgang findet man in der Anleitung genaue Hinweise.

»Paint Magic« wird ab September von Happy Software zum Preis von zirka 80 Mark für den Commodore 64 auf Diskette angeboten.

(Markus Braun/wg)

# ISM 64 — ohne Fleiß kein Preis



Das Auswahlmenü von ISM 64.

**ISM stellt eine Ausnahme unter den Datenverwaltungsprogrammen dar. Es unterscheidet sich in einigen wesentlichen Punkten von Superbase 64, Multidata oder Datamat.**

Im Gegensatz zu den anderen Dateiprogrammen wie Superbase 64 oder Multidata ist ISM 64 kein fertiges Anwendungsprogramm. Es handelt sich vielmehr um eine Systemerweiterung, die, erst eingearbeitet in ein eigenes Programm, richtig nutzbar ist.

Diese Konzeption hat Vor- und Nachteile: Zum einen bietet sie die Möglichkeit der individuellen Problemlösung, zum anderen kann aber insbesondere der Anfänger vor so große programmtechnische Probleme gestellt werden, daß er ISM 64 am Ende gar nicht nutzen kann.

Der hier vorliegende Testbericht geht daher nicht nur auf die Möglichkeiten ein, die ISM 64 bietet, sondern gibt auch Hinweise zur Programmierung.

Grundlage dieses Berichts ist die mehrmonatige Arbeit mit ISM 64 bei der Erstellung eines umfassenden Anwenderprogramms.

## Allgemeines

ISM 64 selbst ist ein Maschinenprogramm von 15 KByte Länge und belegt normalerweise den Basic-Speicher ab dez. 27000. Für das Anwenderprogramm stehen dann noch etwa 24 KByte zur Verfügung. Auch wenn man noch einen gewissen Bedarf an Speicherplatz für Daten abzieht, sollte dies ausreichend sein. Das Maschinensprache-RAM (dez. 49152 bis 53247) bleibt frei; hier lokalisierte Programme (zum Beispiel Interfaces oder Programme zur Unterstützung der Floppy) sollten normal verwendbar sein.

Nach dem Laden von ISM 64 stehen eine Reihe spezieller Befehle zur Einrichtung und Bearbeitung von Dateien zur Verfügung. Diese Befehle sind sehr mächtig und nicht mit jenen zu vergleichen, die man von den üblichen Interpretererweiterungen wie zum Beispiel Simons Basic kennt. Das Schreiben, Ändern und Suchen von Datensätzen wird dann sehr bequem und effektiv.

## Was bietet ISM 64?

ISM 64 beeindruckt zunächst durch die Möglichkeit, praktisch unbegrenzt lange Datensätze verwalten zu können. Ein einzelner Satz darf bis zu 31875 Zeichen lang sein, er kann in bis zu 125 Felder unterteilt werden, wobei jedes Feld bis zu 255 Zeichen enthalten darf. Die Felder können von fester oder variabler Länge sein, bis zu 40 Felder lassen sich als Schlüsselfelder definieren. Angesichts dieser Zahlen fragt man sich natürlich, ob dies überhaupt sinnvoll ist. Welcher Privatanwender hat schon Datensätze mit Zehntausenden von Zeichen oder Dutzenden von Schlüsseln? Hier muß man wissen, daß der ISM nicht speziell für den C 64 entwickelt wurde. Es handelt sich vielmehr um die angepaßte Version eines Programms, das in erster Linie zum gewerblichen Einsatz auf den CBM-Computern der 8000er-Serie gedacht ist. Aber davon abgesehen, auch bei der privaten Anwendung sind Satzlängen von um die 1000 Zeichen durchaus nicht unrealistisch. Wenn man zum Beispiel Literaturdaten verwaltet und jedem Eintrag eine kurze Zusammenfassung der Literaturstelle hinzufügt, sind 500 bis 1000 Zeichen schnell erreicht.

Entsprechend weitgehend sind auch die Möglichkeiten, die dem Anwender bei der Einrichtung der Datei zur Verfügung stehen. Wie bereits erwähnt, kann ein Feld des Satzes als variabel definiert werden, das heißt die Länge des Feldes kann von Satz zu Satz unterschiedlich sein. Hierbei wird auf der Floppy aber immer nur der Platz belegt, der der tatsächlichen Länge des Satzes entspricht. Dies ist keinesfalls selbstverständlich, andere Systeme sind da nicht so flexibel. Dem Anwender bleibt auch überlassen, welche Felder er wirklich mit Daten füllen will. Auf diese Weise ist es möglich, ohne Speicherplatzverschwendung den Datensatz großzügig zu definieren und erst nach und nach alle Felder auszufüllen. Ein Datensatz kann so im Laufe der Zeit dynamisch wachsen.

Bis zu 40 Felder eines Satzes können als Schlüssel definiert werden.

# ISM 64 — ohne Fleiß kein Preis

Von diesen Feldern werden bis zu 48 Zeichen linksbündig in die Schlüsseldatei eingetragen. Bei der Suche nach einem Satz wird dann zunächst auf diese Datei zugegriffen (siehe unten). Bei ISM 64 können die Schlüssel als ein- oder mehrdeutig definiert werden. Ein mehrdeutiger Schlüssel (zum Beispiel Nachname) darf mehrfach vorkommen, durch sequentielle Suche (siehe unten) können dann alle Sätze mit diesem Schlüssel gefunden werden. Hat man dagegen einen Schlüssel als eindeutig definiert (zum Beispiel sinnvoll bei einer laufenden Nummer), dann weist der Dateiverwalter diesen Satz mit einer entsprechenden Fehlermeldung zurück. Der Satz geht hierbei natürlich nicht verloren, sondern kann vom Anwenderprogramm zur Korrektur weiterverarbeitet werden. Überhaupt ist die Fehlerbehandlung bei ISM 64 sehr gelungen. Systemabstürze lassen sich bei geschickter Programmierung weitgehend ausschließen.

An dieser Stelle erscheint es sinnvoll, etwas ausführlicher auf die Therorie der Dateiverwaltung einzugehen. ISM 64 verwendet hier keine spezielle Lösung, sondern verwaltet die Sätze nach dem Prinzip der relativen Datei, wobei Schlüssel und Sätze in unterschiedlichen Dateien abgelegt werden. Relativdateien werden vom Basic V 2.0 ja leider nicht unterstützt, und sind daher sonst nur durch Kunstgriffe zugänglich.

## Das Prinzip der relativen Datei

Wie ist nun eine relative Datei aufgebaut? Das Prinzip ist so einfach wie wirkungsvoll: Der Speicherplatz auf der Floppy wird zunächst in kleine Häppchen, die sogenannten Records, zerlegt. Die Länge der einzelnen Records ist bei ISM 64 frei wählbar. Dies ist wichtig, wenn man den bei der Floppy VC 1541 ja nicht gerade üppig bemessenen Speicherplatz optimal nutzen will. Die einzelnen Records werden intern durchnumeriert. Ein Datensatz wird nun in diese Records geschrieben, ein Satz von beispielsweise 300 Byte Länge belegt dann bei einer gewählten Recordlänge von 100 Byte gerade drei Records. Dem Satz wird dann vom Programm intern zur eindeutigen Unterscheidung von anderen Sätzen die Nummer zugewiesen, die der Nummer des ersten belegten Records entspricht. Gleichzeitig werden die Schlüssel des Satzes — dies sind die Einträge, über die man später auf den Satz zugreifen möchte — in eine eigene Datei, die Schlüsseldatei, in einen sogenannten Schlüsselbaum eingetragen. Die Schlüssel werden von vornherein alphabetisch beziehungsweise numerisch sortiert. Bei jedem Schlüssel findet sich dann als Verweis auf den zugehörigen Satz die Nummer des Records, mit dem der Satz beginnt. Diese Nummer ist naturgemäß umso höher, je weiter der Satz relativ vom Anfang der Datei entfernt ist, je mehr Records zum Zeitpunkt des Eintrags also schon belegt waren. So erklärt sich die Bezeichnung Relativdatei. Bei der Suche nach einem bestimmten Satz sucht der Dateiverwalter im Schlüsselbaum nach dem angegebenen Suchbegriff. Ist dieser vorhanden, dann merkt sich der Dateiverwalter die beim Schlüssel eingetragene Recordnummer. Diese gibt die relative Lage des entsprechenden Satzes zum Dateianfang an, und der Dateiverwalter kann auf diesen Satz zugreifen. Das Prinzip der Datenspeicherung in Relativdateien gilt momentan als optimal bei der Arbeit mit Kleincomputern.

Wie bereits erwähnt, stellt ISM 64 Befehle zum Eintragen, Ändern und Suchen von Datensätzen zur Verfügung. Der Aufruf der einzelnen Befehle erfolgt aus einem ganz normalen Basic-Programm heraus, der Komfort der Arbeit ist in erster Linie von der Qualität des Anwenderprogramms abhängig. Die Befehle an sich lassen in bezug auf die dem Anwendungsprogrammierer gegebenen Möglichkeiten eigentlich kaum Wünsche offen.

**Datensätze eintragen**

Jeder Datensatz kann und sollte unmittelbar nach der Erstellung auf die Floppy geschrieben werden, um nicht das Risiko eines etwaigen Datenverlustes, beispielsweise infolge eines Stromausfalls, einzugehen. Die Zeit, die zum Eintrag benötigt wird, steigt mit der Anzahl der Schlüssel stark an. Allerdings werden die Schlüssel beim Eintrag bereits alphabetisch sortiert, ein Arbeitsgang, der bei der Ausgabe von sortierten Listen bei den meisten anderen Programmen später bewältigt werden muß. Vierzig Schlüssel sind bei der bekannten »Schnelligkeit« der VC 1541 sicher kaum realisierbar. ISM 64 bietet hier allerdings die Möglichkeit der Stapelverarbeitung, das heißt man trägt zunächst nur den Satz ein. Dies geht sehr schnell, der zeitaufwenige Eintrag der Schlüssel in den sortierten Baum erfolgt erst später. Dies kann durch ein kleines Programm geschehen, ohne daß man dabei sein muß. Die Realisierung dieser Möglichkeit, Satz und Schlüssel getrennt voneinander einzutragen, ist allerdings eine Sache des Anwenderprogramms. Auf diese Weise ist es dann möglich, zu einem Satz mehr

```
Ausdruck von Datensätzen
------------------------
1000 dim dn$(1):dn$(1)="Dateiname"
1010 !iopen,10,dn$(1),5:rem Datei öffnen
1020 !iread,m,1,df$(1),rn,fe:rem Datensatz lesen
1030 !iclose:rem Datei schließen
1040 open4,4:rem Druckerkanal öffnen
1050 for i=1 to n
1060 print#4,df$(i)
1070 next:rem Felder des Satzes drucken
1080 close4:rem Druckerkanal schließen
1090 open15,8,15:rem Befehlskanal der Floppy öffnen
1100 print#15,"i":rem Floppy initialisieren
1110 close15:rem Befehlskanal der Floppy schließen
1120 return:rem z.B. nächsten Satz lesen
```

**Nach jedem Ausdruck muß die Floppy initialisiert werden. Diese kleine Routine gibt auch einen Eindruck, wie die ISM-Befehle in ein Basicprogramm eingebaut werden.**

Schlüssel einzutragen, als bei der Definition der Datei vorgesehen war.

**Datensätze ändern**

Das Ändern eines Satzes ist möglich, indem man den Satz zunächst liest, die Änderungen vornimmt, und den Satz dann mit einem speziellen Schreibbefehl wieder einträgt. Auch hier ist der Komfort von der Qualität des Anwenderprogramms abhängig. Zusätzlich ist es möglich, einen Satz komplett mit allen Schlüsseln, nur den Satz, nur die Schlüssel, oder aber nur einzelne Schlüssel des Satzes zu löschen (und gegebenenfalls durch neue zu ersetzen).

**Datensätze suchen**

Sicher am wichtigsten bei einem Dateiverwaltungssystem sind die Möglichkeiten, die man zur Suche nach Datensätzen hat. ISM 64 bietet hier Befehle zur Suche mit voller Übereinstimmung eines Schlüssels mit einem Suchbegriff, sowie zur Suche mit verkürztem Schlüssel (zum Beispiel Suche nach Sätzen, bei denen ein Schlüssel mit A beginnt). Zusätzlich kann über die Recordnummer direkt auf einen bestimmten Satz zugegriffen werden. Eine Suche mit Joker (zum Beispiel nach einem »M??er« ist nicht von vornherein vorgesehen, kann aber im Anwenderprogramm leicht realisiert werden. Außerdem erlaubt ISM die sequentielle Suche, das heißt, man liest den ersten Satz, auf den ein bestimmtes Kriterium zutrifft, und kann dann alle weiteren Sätze lesen. Diese Option erlaubt auch das schnelle Durchblättern der Datei. Oft findet man so einen interessierenden Satz am schnellsten. Das Zeitverhalten bei der Suche ist noch befriedigend, die Grenzen der Floppy VC 1541 werden hier mehr als offenbar. Ein kleiner Nachteil ist sicherlich, daß vom Programm Groß- und Kleinschreibung unterschieden werden. Wenn man nach einem Mayer mit mayer sucht, dann wird der entsprechende Satz nicht gefunden.

**Drucken von Datensätzen**

Das Drucken von Datensätzen wird von ISM 64 nicht unterstützt, die Druckroutine ist allein Sache des Anwenderprogramms. Dies sollten insbesondere alle C 64-Anwender, die keinen Commodore-Drucker haben, nicht bedauern. Oft muß man ja feststellen, daß bei fertigen Programmlösungen andere als Commodore-Drucker, wenn überhaupt, nur unvollständig unterstützt werden. Die Möglichkeit einer ganz individuellen Druckroutine ist in solchen Fällen eher ein Vorteil. Allerdings muß man bei der Programmierung einiges unbedingt beachten (siehe unten).

## Die Programmierung von ISM 64

Wie bereits oft erwähnt, ist ISM nur zusammen mit einem Anwenderprogramm voll nutzbar. Wie einfach oder schwierig ist nun die Erstellung eines solchen Programms? Ein möglicher Erfolg beziehungsweise Mißerfolg hängt erstens davon ab, wie gut man das Basic V 2.0 beherrscht und zweitens, wie hoch die eigene Frustrationsschwelle liegt. Wenn man gut programmieren kann, und nicht so schnell aufgibt, wird man sicherlich zu einer befriedigenden Lösung kommen. Das mitgelieferte Handbuch ist umfassend, auch im Detail. Allerdings ist es nicht einfach zu lesen. Viele Kapitel wird auch der Fortgeschrittene mehrfach studieren müssen, vieles wird auch erst im Laufe der Arbeit mit dem Programm klar werden. Hilfreich ist es, daß ein fertiges, kommentiertes Beispielprogramm auf der Diskette mitgeliefert wird. Dieses Programm kann normal gelistet werden, und ist durchaus nützlich zur Inspiration bei der Programmerstellung. Leider enthält es mindestens einen programmtechnischen Fehler. Ein anderes, mitgeliefertes Programm erlaubt die Anlage beliebiger Dateien. Diese Routine ist sehr brauchbar, so daß keine Notwendigkeit besteht, eine eigene Anlage-Routine zu schreiben.

Wie bereits eingangs ausgeführt, liegt eine der Hauptstärken des ISM 64 darin, praktisch unbegrenzt lange Datensätze verwalten zu können. Unter anderem darf ein einzelnes Feld bis zu 255 Zeichen lang sein. Leider ist im Basic 2.0 eine adäquate Eingabemöglichkeit nicht vorgesehen. Das INPUT-Statement erlaubt nur Eingaben bis zu real 78 Zeichen, wobei keine Kommata vorkommen dürfen. Einziger Ansatzpunkt ist das GET-Statement.

Ein besonderes Kapitel ist der Ausdruck von Datensätzen. Dies ist wie gesagt allein Sache des Anwenderprogramms. In einem solchen Programm ist nun folgendes zu berücksichtigen: Nach jedem Ausdruck muß die Floppy initialisiert werden. Wird die Floppy nach einem Ausdruck nicht initialisiert, dann führt der nächste Dateizugriff zu einem Systemabsturz. Da dieser Effekt auftrat:

1. Bei einem über ein Software-Interface am User-Port angeschlossenen Epson-Drucker
2. bei einem normal angeschlossenen VC 1526 und
3. auch wenn überhaupt gar kein Drucker angeschlossen war, scheint es sich um ein grundsätzliches Problem zu handeln, dessen Ursache wohl in der Unzulänglichkeit des seriellen Busses liegt. Wie programmtechnisch beim Ausdruck verfahren werden kann, ist aus dem abgedruckten Listing ersichtlich. Leider wird in der ISM-Anleitung nichts von dieser Problematik erwähnt, obwohl dies meiner Ansicht nach unbedingt notwendig wäre. Denn was nutzt einem letztendlich ein Datenverwaltungsprogramm, wenn man nicht drucken kann.

## Fazit

ISM 64 ist ein sehr leistungsfähiges Programm mit professioneller Konzeption. Die Notwendigkeit, ein eigenes Anwenderprogramm schreiben zu müssen, kann ein Vorteil oder Nachteil sein, je nach Art des geplanten Einsatzes und den Kenntnissen des Anwenders. Der Preis von zirka 140 Mark ist sicherlich günstig zu nennen.

Am Ende sei noch erwähnt, daß für alle, die ISM 64 einsetzen möchten, — ohne ein eigenes Anwenderprogramm zu schreiben — die Möglichkeit besteht, ein fertiges Programm mit Anleitung vom Autor dieses Berichtes zu beziehen. Der Autor ist auch gerne bereit, anderen ISM-Anwendern bei aufgetretenen Problemen im Rahmen seiner Möglichkeiten weiterzuhelfen. Schreiben sie an H. J. Schlicht, Kirschgartenstr. 7, 6900 Heidelberg 1

(H. J. Schlicht/rg)




Der heiße Kampf vor dem Korb bei »One on One«

**One on One wurde als das erste und heißeste Basketballprogramm angekündigt, das die Fans von Sportspielen je gesehen haben. Wir haben uns »One on One« einmal näher betrachtet. Getestet wurde es auf dem Commodore 64.**

Nun zum Spiel (Preis zirka 129 Mark) selbst. Nachdem man das Programm von Diskette (es gibt keine Kassettenversion) in den C 64 geladen hat, ertönt zunächst die mehrstimmige Anfangsmelodie, und die beiden Hauptakteure Julius Erwing und Larry Bird demonstrieren gekonnt auf dem Bildschirm das Spiegelgeschehen (siehe Bild). Betätigt man den Feuerknopf, kommt man in das Auswahlmenü. Im Gegensatz zu vielen anderen (Sport-) Spielen können hier sehr viele Parameter vom Spieler selbst eingestellt werden, auch wenn das eigentliche Spielgeschehen gleich bleibt.

## Lassen Sie sich nicht austricksen

Folgende Details können verändert werden:

1. Anzahl der Spieler (ein Spieler gegen den Computer oder zwei Spieler gegeneinander)
2. Spielstärke des Computer-Gegners (vom Anfänger bis Profi)
3. Spielart:
a) Spiel auf Zeit (2 bis 8 Minuten pro Viertel)
b) Spiel auf Erreichen einer gewissen Punktzahl (ebenso vom Spieler einstellbar)
4. Festlegung, welcher Spieler nach erfolgreichem Korbwurf den Ball erhält.

Nützlich für den Spielablauf ist es ebenfalls, daß man während des Spiels jederzeit in das Auswahlmenü zurückkehren kann, um gegebenenfalls einige Änderungen vorzunehmen und das unterbrochene Spiel dann fortzusetzen. Nachdem man dies alles überstanden hat, kann man anfangen zu spielen. Es handelt sich bei One on One nicht um ein Spiel zweier Mannschaften auf zwei Körbe, sondern um ein Spiel Mann gegen Mann auf einen Korb. Doch nichtsdestotrotz erlebt man nun eine (fast) perfekte Simulation des Basketballspiels. Die Bewegungen der Spieler, zum Beispiel beim Sprungwurf, sind täuschend den echten Bewegungen der Spieler im Basketballsport nachempfunden. Auch die sonstigen Feinheiten entsprechen den Original-Basketballregeln (bis auf einige Ausnahmen). So besteht die Möglichkeit, den Gegner zu »foulen«; in diesem Fall erscheint der Schiedsrichter auf dem Bildschirm und pfeift das Spiel ab. Je nach Art des Fouls wird einem zum Beispiel ein Freiwurf zugesprochen. Eine weitere Besonderheit ist, daß man den Korb »zerschmettern« kann. Die heruntergefallenen Teile werden dann vom Hilfspersonal aufgekehrt. Der Spielstand, die noch verbleibende Spielzeit und die Anzahl der von den Spielern begangenen Fouls werden am oberen linken Bildschirmrand angezeigt. Der Kraftverlust, der zum Beispiel durch dauerndes Springen eintritt, wird mittels zwei waagrechten Balken am unteren Rand des Bildschirms dargestellt.

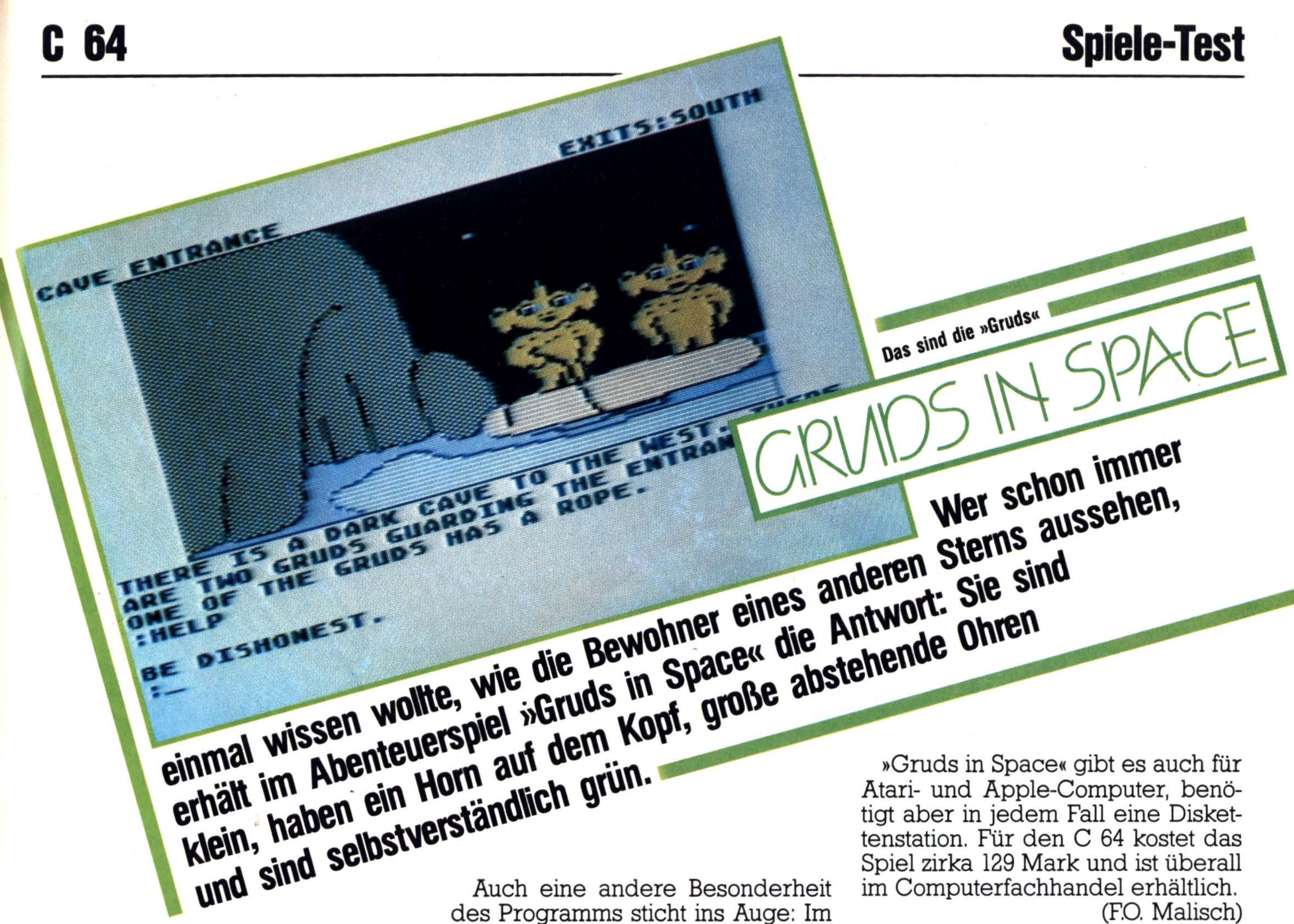
Je größer der Kraftverlust ist, desto niedriger springt der jeweilige Spieler. Dies ist vor allem beim Verhindern eines Korbwurfs des Gegners von Bedeutung.

Die Regelung, daß ein Spieler den Ball nur 24 Sekunden lang ohne versuchten Korbwurf führen darf, ist für einen flüssigen Spielablauf sehr wichtig. So kommt es fast durchgehend zu packenden Szenen vor dem Korb. Ein weiteres wichtiges Detail, das für das Spiel von sehr großer Bedeutung ist, ist die Möglichkeit, durch kurzen Druck auf den Feuerknopf den Spieler um seine eigene Achse drehen zu lassen. Obwohl dieses Programm von den Softwarekünstlern »Electronic Arts« stammt, die für ihre perfekten Programme bekannt sind, gibt es auch einige Punkte zu bemängeln. So kann zum Beispiel die Weite des Wurfs nicht beeinflußt werden. Der Ball bewegt sich automatisch in Richtung Korb, wobei die Wahrscheinlichkeit eines Treffers von der Entfernung zum Korb abhängt.

## Ein fast perfektes Programm

Trotz einiger weniger Kritikpunkte muß man diesem Spiel eine sehr hohe Qualität bescheinigen. Elektronic Arts ist es wieder einmal gelungen, eine gute Spielidee in ein (fast) perfektes Programm umzusetzen. Obwohl nur auf einen Korb gespielt wird, bereitet dieses Programm dem Benutzer eine lang andauernde Spielfreude, da, und dies ist sehr wichtig, der Erfolg größtenteils vom Können des Spielers abhängt.

(Martin Gaksch/gk)




Das sind die »Gruds«

# Gruds in Space

**Wer schon immer einmal wissen wollte, wie die Bewohner eines anderen Sterns aussehen, erhält im Abenteuerspiel »Gruds in Space« die Antwort: Sie sind klein, haben ein Horn auf dem Kopf, große abstehende Ohren und sind selbstverständlich grün.**

Bei »Gruds in Space« übernehmen Sie als Spieler die Rolle eines Raumschiffkommandanten, von dessen Geschick die Zukunft der Menschheit abhängt. Nach Meinung des Oberkommandos der Vereinigten Galaktischen Flotten sind Sie als einziger in der Lage, unsere im All gestrandeten Kriegsschiffe mit lebenswichtigem Treibstoff zu versorgen. Gelingt dies nicht, so ist der Niedergang unserer Zivilisation gewiß.

Das allein sollte Grund genug sein, diesen schwierigen Auftrag anzunehmen. Er führt uns auf eine Reise durch das Sonnensystem in deren Verlauf es viele knifflige Rätsel zu lösen gilt. Wie bereits im Titelbild des Spiels zu erkennen, werden wir dabei auch die Bekanntschaft der schon sprichwörtlichen kleinen grünen Männchen aus dem Weltall — der sogenannten »Gruds« — machen. Es ist selbstverständlich mehr als ein Zufall, daß eben diese »Gruds« zugleich Werbemaskottchen von Sirius Software sind.

Obwohl man eine derartige Schleichwerbung als eher störend empfinden könnte, ist hier genau das Gegenteil der Fall. Schon allein durch ihre urkomische äußere Erscheinung vermögen die »Gruds« viel zum Spielwitz dieses Abenteuers beizutragen.

Auch eine andere Besonderheit des Programms sticht ins Auge: Im Gegensatz zu den bisher üblichen starren Grafiken wurden bei »Gruds in Space« die Bilder, die den Spieler durch die Handlung führen, mit Bewegungs- und Toneffekten angereichert. So kann man zum Beispiel direkt sehen, wie ein angriffslustiges Monster siegessicher zwinkert oder wie in einem Sumpfgebiet Gasblasen aus dem Boden steigen. Aufgrund der aufwendigen Grafik benötigt das Programm aber auch sehr wenig erklärenden Text, so daß man selbst mit bescheidenen Englischkenntnissen selten auf ein Wörterbuch zurückgreifen muß.

»Gruds in Space« gibt es auch für Atari- und Apple-Computer, benötigt aber in jedem Fall eine Diskettenstation. Für den C 64 kostet das Spiel zirka 129 Mark und ist überall im Computerfachhandel erhältlich.

(F.O. Malisch)

## Druckfehlerteufelchen

Betrifft: Namensgebung für den Druckfehlerteufel

Eine recht aussichtlose Sache! Der Druckfehler ist nämlich ein sehr wandelbares Wesen. Kaum hat man ihn in Gestalt eines Dreckfuhlers erwischt, kommt einem dieses Fuhrdreckel in der Verkleidung als Dreckhufler wieder aus. Oft wird das Durchferkel auch zur Furchdrelke.

Der Ulkferch bietet sich scheinheilig als Druckhelfer an und wird dabei unversehens zum Druckhefler, dieser Krechdeufl.

Gelegentlich versteckt sich der Drehflucker hinter dem Namen Krechfluder.

In Wahrheit aber ist der Frechdruckel aber ein ganz normaler Hufdreckler.

Mit rfuendiclhne Gürssne!

PS: Entschuldigen Sie bitte meine Fluckdreher, wollte drucken: meine Herfdruckel!

Giselbert Kosmala

# Los Angeles läßt grüßen




Wasser hat keine Balken: Ein Turmspringer macht sich bere

So wird's gemacht: Eine glatte Landung bei der Gymnastik

**Wenn Sie sich »aktiv« an den olympischen Wettkämpfen beteiligen wollen: »Summer Games« macht's möglich.**

Alle Welt blickt in diesen Tagen gespannt nach Los Angeles, wo die olympischen Sommerspiele stattfinden. Dieses sportliche Großereignis hat sich auch auf die Spiele-Szene für den Commodore 64 ausgewirkt. Das amerikanische Software-Haus »Epyx« präsentiert mit »Summer Games« ein Sport-Spiel, das eine Goldmedaille verdient.

Am Anfang stehen zwei Gags: Nach Einladen des Programms erfreut eine farbenprächtige Eröffnungszeremonie das Auge des Zuschauers, die einen Vorgeschmack auf die durchgehend hervorragende Grafik von »Summer Games« liefert. Bevor die Medaillen-Hatz losgeht, darf sich jeder Spieler noch eines von 16 Ländern heraussuchen, woraufhin die entsprechende Nationalhymne aus dem Lautsprecher scheppert.

Nun geht's richtig los: Nachdem der Computer ein knappes Minütchen nachgeladen hat, treten die Tele-Athleten zum Stabhochsprung an. Der Anfänger braucht hier schon eine Menge Übung, um unter dem Jubel der Zuschauer die Stange erfolgreich zu überqueren. Timing ist hier alles. Doch auf zur nächsten Disziplin: »Platform Diving« ist angesagt. Mehr oder weniger elegant segeln die Turmspringer ins Wasser, wobei man durch entsprechende Joystick-Bewegungen vier verschiedene Kunstfiguren verursachen kann. Produziert man eine Bauchlandung, vergibt der Computer schon mal eine 0,0 als Note. Kondition ist bei der 400 Meter Staffel gefragt: Hier muß man seine Kräfte, die in Form eines Balkens am unteren Bildschirmrand dargestellt werden, gut einteilen. Wilder geht es beim 100 Meter-Lauf zu. Wer den Joystick am schnellsten im Kreise dreht, kann eine Zeit unter 10 Sekunden erreichen.

Nach den Laufwettbewerben geht's dann mit der Gymnastik weiter. Hier ist viel Training erforderlich, um eine junge Dame über ein Pferd hüpfen zu lassen, ohne daß sie bei der Landung auf die Nase fällt, was bei den Mitspielern in der Regel ein hämisches Grinsen hervorruft. Bei den beiden Schwimm-Wettbewerben ist dann wieder Timing gefragt, da die Schwimmer nur Tempo machen, wenn man den Feuerknopf in regelmäßigen Intervallen betätigt. Abschließend ist Tontaubenschießen angesagt. Umrahmt von einer reizvollen Landschaft müssen möglichst viele von 25 umherzischenden Tontauben getroffen werden. Zu guter Letzt gibt der Computer den Endstand bekannt und spielt noch einmal die Hymne des Olympiasiegers.

»Summer Games« ist eine rundum gelungene Heim-Olympiade, die mich spontan begeistert hat. Eine Diskette (Preis: zirka 109 Mark), die man auch nach vielen Stunden gerne wieder spielt.

(Heinrich Lenhardt)




**Nachdem wir sie in den ersten vier Folgen kaum erwähnt haben, sind sie heute dran: die Sprites. Sie müssen einfach in jeder Reise durch die Grafik unseres C 64 auftauchen.**

Sprite heißt auf deutsch soviel wie »Kobold«, »Gespenst«. Wie richtige Kobolde können sie sowohl in Dornröschens Schloß (also im Bit-Map-Modus) als auch außerhalb (nämlich im Normalmodus) über den Bildschirm geistern — anscheinend dabei allen bisher gelernten Regeln über die Grafik widersprechen. Wir werden in dieser Folge zwar lernen, mit ihnen umzugehen, sogar sie zu beherrschen — aber ihre genaue Funktion und Herkunft wird weiterhin im dunkeln bleiben: Meines Wissens gibt es noch kein Listing des Spriteunterstützenden Maschinenprogramms, das wohl im tiefsten Dunkel des VIC-II-Chip verborgen liegt: Denn der VIC-II-Chip belegt ja die Speicherplätze 53248 bis 54271. Im erreichbaren Teil dieses Zauberschlosses (53248 bis 53294) liegen die bisher viel von uns begangenen 47 Register (siehe 1. Folge, Tabelle 1), aber wo ist die Geheimtür zu den anderen 978 Bytes? Alles recht romantisch, werden Sie sagen. Nun — ganz so mysteriös ist die Sache nun auch wieder nicht, wie uns der Name Sprite = Kobold, Gespenst einreden will. Gehörig entschleiert wird das Geheimnis schon durch den anderen englischen Ausdruck für diese Dinger: MOBs. Das bedeutet: Movable Object Blocks, also bewegliche Blöcke von Objekten (Bildern, Darstellungen). Sie werden sehen, wenn wir mit dem MOBs umgehen können, ist das verbleibende Geheimnis eigentlich keines mehr, sondern verwandelt sich nur noch in eine Herausforderung für einen Maschinensprache-Programmierer.

Wir werden zunächst einmal schöpferisch tätig und erschaffen ein Sprite. Der erste Schritt dazu ist kreativ: Wie soll das Ding aussehen? Da bietet sich ja zum Beispiel der aus der Magie bekannte Drudenfuß an, auch Pentagramm genannt, weil wir ja schließlich diese Sprite-Geister bannen wollen (siehe Bild 1).

Jetzt müssen wir diese Zeichnung in eine Form bringen, die unser Computer versteht, also in Bytes. Wie auch schon bei den Buchstaben und der hochauflösenden Grafik sind hier wieder gesetzte und gelöschte Bits in den Bytes die Anzeige für »Punkt sichtbar« oder »Punkt nicht sichtbar«. Das im VIC-II-Chip organisierte Sprite-Programm nimmt die Bytes in der im Bild 2 gezeigten Anordnung wahr.

Den so gebildeten Block verwaltet es in genau dieser Anordnung als ein Sprite. Deswegen müssen wir uns nun die Mühe machen, unser Pentagramm in so ein Bit-Raster ein-

| Zeile | Byte | binär | dez. | Zeile | Byte | binär | dez. |
|---|---|---|---|---|---|---|---|
| | 0 | 00000000 | 4 | | 33 | 00000001 | 1 |
| 1 | 1 | 00011000 | 24 | 12 | 34 | 10000001 | 129 |
| | 2 | 00000000 | 0 | | 35 | 10000000 | 128 |
| | 3 | 00000000 | 0 | | 36 | 00000001 | 1 |
| 2 | 4 | 00011000 | 24 | 13 | 37 | 11000011 | 195 |
| | 5 | 00000000 | 0 | | 38 | 10000000 | 128 |
| | 6 | 00000000 | 0 | | 39 | 00000001 | 1 |
| 3 | 7 | 00111100 | 60 | 14 | 40 | 10111101 | 189 |
| | 8 | 00000000 | 0 | | 41 | 10000000 | 128 |
| | 9 | 00000000 | 0 | | 42 | 00000011 | 3 |
| 4 | 10 | 00100100 | 36 | 15 | 43 | 00011000 | 24 |
| | 11 | 00000000 | 0 | | 44 | 11000000 | 192 |
| | 12 | 00000000 | 0 | | 45 | 00000011 | 3 |
| 5 | 13 | 01100110 | 102 | 16 | 46 | 00111100 | 60 |
| | 14 | 00000000 | 0 | | 47 | 11000000 | 192 |
| | 15 | 00000000 | 0 | | 48 | 00000110 | 6 |
| 6 | 16 | 01100110 | 102 | 17 | 49 | 11000011 | 195 |
| | 17 | 00000000 | 0 | | 50 | 01100000 | 96 |
| | 18 | 00000000 | 0 | | 51 | 00000111 | 7 |
| 7 | 19 | 01100110 | 102 | 18 | 52 | 10000001 | 129 |
| | 20 | 00000000 | 0 | | 53 | 11100000 | 224 |
| | 21 | 00111111 | 63 | | 54 | 00000110 | 6 |
| 8 | 22 | 11111111 | 255 | 19 | 55 | 00000000 | 0 |
| | 23 | 11111100 | 252 | | 56 | 01100000 | 96 |
| | 24 | 00011000 | 24 | | 57 | 00001100 | 12 |
| 9 | 25 | 01000010 | 66 | 20 | 58 | 00000000 | 0 |
| | 26 | 00011000 | 24 | | 59 | 00110000 | 48 |
| | 27 | 00001100 | 12 | | 60 | 00000000 | 0 |
| 10 | 28 | 11000011 | 195 | 21 | 61 | 00000000 | 0 |
| | 29 | 00110000 | 48 | | 62 | 00000000 | 0 |
| | 30 | 00000110 | 6 | | | | |
| 11 | 31 | 10000001 | 129 | | | | |
| | 32 | 01100000 | 96 | | | | |

**Tabelle 1. Kennzahlen für die Berechnung von Sprites**




zufügen. Das ist in Bild 3 geschehen.

Nun muß dieses Bild in einen Zahlencode übersetzt werden. Überall dort, wo ein Bitfeld ausgefüllt ist, steht bei der Binärzahl eine 1, sonst eine Null. Demnach ergeben sich die Kennzahlen in der Tabelle 1.

Dieser Wust an Zahlen legt also unser Sprite fest. Diese doch recht aufwendige Rechnerei und Planerei ist, wenn sie aufmerkamer Leser von Computerliteratur sind — meist nicht mehr nötig. Obwohl es sicher gut ist, das Planen eines MOBs auch von Hand zu beherrschen (wie wir jetzt!), kann man sich die Arbeit doch mächtig erleichtern durch Eintippen eines der vielen Sprite-Editor-Programme, die es in fast allen Fachzeitschriften wie Sand am Meer gibt. Weil hier nicht der 1001. Sprite-Editor abgedruckt werden soll, dient das anliegende Programm »Sprity« anderen Zwecken. (Ein nettes kurzes Listing von H. Kunz finden Sie zum Beispiel in der Zeitschrift Computer persönlich Nr. 21, 1983 auf Seite 120). Jetzt müssen wir noch dafür sorgen, daß der C 64 diese Zahlen irgendwo zugreifbar hat, mit anderen Worten: Sie müssen in den Speicher gePOKEd werden. Im allgemeinen verwendet man dazu eine kleine FOR-NEXT-Schleife in der die in DATA-Zeilen abgelegten Zahlen gelesen und eingePOKEd werden. Wohin packt man die Kennzahlen? Im Prinzip kann man sie überall — wo sie nicht gerade lebenswichtige Computerfunktionen oder Basicprogramme stören — unterbringen. Es gibt lediglich zwei Dinge, die zu beachten sind:

a) Die Startadresse muß durch 64 glatt teilbar sein. (Zum Beispiel 896 = 14 mal 64 etc.)

b) Aus Gründen, auf die wir noch zu sprechen kommen werden, sollten die Sprite-Daten im gleichen 16 KByte-Speicherabschnitt (siehe vorangegangene Folge) abgelegt werden, in dem sich das Video-RAM befindet, welches bei der Sprite-Nutzung angeschaltet ist.

Damit gibt es im Prinzip 256 Orte pro Speicherabschnitt, in denen die Sprite-Daten gespeichert werden können. Allerdings sind einige Stellen zum Beispiel im Abschnitt 0 (mit dem normalen Bildschirm) bei der Verwendung von nur wenigen Sprites besonders bevorzugt, weil man keinen Basicspeicher wegnimmt und deshalb auch keine Schutz-POKEs nötig sind:

1) 704 — 767 ungenutzte Adressen
2) 832 — 895 Kassettenpuffer
3) 896 — 959 Kassettenpuffer
4) 960 — 1023 restlicher Kassettenpuffer und freier Platz.

Für weitere MOBs muß dann Basicspeicher verwendet und dieser dann vor dem Überschreiben durch Programmtext, Variablen oder Strings — wie in Folge 2 gezeigt — geschützt werden. Dem Erfindungsreichtum sind allerdings keine Grenzen gesetzt. So könnte man beispielsweise (siehe vorangegangene Folge) den Bildschirm an den oberen Rand des Abschnittes 2 verschieben und darunter oder darüber Sprite-Daten ablegen. Man muß dann zwar auch den Speicherschutz einPOKEn, hat aber trotzdem meistens noch mehr als genug Basicspeicherplatz. Wenn man nur vier Sprites gleichzeitig verwendet, insgesamt aber mehrere definiert hat, kann man sie alle im 4 KByte-Bereich ab $C000 abspeichern und bei Bedarf den Sprite, der dran ist, mit einer kleinen FOR-NEXT-Schleife in einen der vier Speicherbereiche (704 .... und so weiter) umladen. Sicher fallen Ihnen noch mehr Möglichkeiten ein. Wir werden uns im nachfolgenden einfach mit drei Sprites begnügen und uns nur im normalen Abschnitt 0 bewegen.

Legen wir also nun zunächst unser Pentagramm in den Bereich 704 bis 767 und nach 832 bis 895:

```
10 FOR I=0 TO 62:READ A:POKE
704+I,A:POKE 832+I,A:NEXT I
30 DATA hier werden jetzt unsere 63
Kennzahlen eingegeben.
bis 60 DATA
```

| Sprite gerade noch voll sichtbar | | Sprite gerade nicht mehr sichtbar | |
|---|---|---|---|
| links oben | rechts oben | links oben | rechts oben |
| X = 24<br>Y = 50 | X = 320<br>Y = 50 | X = 0<br>oder/und<br>Y = 29 | X = 344<br>oder/und<br>Y = 29 |
| links unten | rechts unten | links unten | rechts unten |
| X = 24<br>Y = 229 | X = 320<br>Y = 229 | X = 0<br>oder/und<br>Y = 250 | X = 344<br>oder/und<br>Y = 250 |

Tabelle 2. Grenzposition normaler Sprites

| Sprite No. | Register |
|---|---|
| 0 | 53287 |
| 1 | 53288 |
| 2 | 53289 |
| . | . |
| . | . |
| . | . |
| 7 | 53294 |

Tabelle 3. Zuordnung der Sprite-Farben-Register

| Bit-Paar | Farbherkunft |
|---|---|
| 00 | durchsichtig |
| 01 | Sprite Mehrfarbregister 53285 |
| 10 | normale Sprite-Farben-Register (53287 bis 53294) |
| 11 | Sprite Merhfarbregister 52286 |

Tabelle 4. Herkunft der Bitpaar-Farben im Multicolor-Modus

| Sprite gerade noch voll sichtbar | | Sprite gerade nicht mehr sichtbar | |
|---|---|---|---|
| links oben | rechts oben | links oben | rechts oben |
| X = 24<br>Y = 50 | X = 296<br>Y = 50 | X nicht möglich<br>nur:<br>Y = 8 | X = 344<br>oder/und<br>Y = 8 |
| links unten | rechts unten | links unten | rechts unten |
| X = 24<br>Y = 208 | X = 926<br>Y = 208 | X nicht möglich<br>nur:<br>Y = 250 | X = 344<br>oder/und<br>Y = 250 |

Tabelle 5. Grenzposition doppelt gedehnter Sprites

Noch ignoriert der C 64 unsere Sprites völlig. Es interessiert ihn überhaupt nicht, was wir an Arbeit zum Füllen seiner Speicherbauches aufgewendet haben. Wir müssen ihm noch mitteilen, wo der VIC-II-Chip die Sprite-Daten finden kann. Weil dieser Chip dazu eingerichtet ist, gleichzeitig 8 MOBs zu verwalten, gibt es acht Speicherzellen, die sogenannte Sprite-Zeiger enthalten.

## Na, wo ist er denn? Sprite-Zeiger

Sie befinden sich immer im gleichen 1 KByte-Bereich, in dem auch der Bildschirmspeicher liegt. Weil dieser Video-RAM nur 1 000 Bytes benötigt, sind oberhalb desselben noch 24 Bytes frei, von denen die obersten 8 als Sprite-Zeiger dienen. Wenn also im Normalfall der Bildschirmspeicher von 1024 bis 2023 geht, dann liegen die Sprite-Zeiger von 2040 bis 2047. Verschiebt man den Bildschirm, dann werden die Sprite-Zeiger mit verschoben. In diese Sprite-Zeiger-Bytes POKEn wir die Zahlen ein, die mit 64 multipliziert die Startadresse der Sprite-Daten ergeben. In unserem Beispiel also: 704/64 = 11 und 832/64 = 13.

Wenn wir diese Sprite-Zeiger eingeben, nehmen wir automatisch gleichzeitig auch die Numerierung vor. Dabei gehört zu
Sprite Nr. 0 der Sprite-Zeiger 2040
Sprite Nr. 1 der Sprite-Zeiger 2041
Sprite Nr. 2 der Sprite-Zeiger 2042
und so weiter.

Nennen wir also einfach den Sprite, dessen Daten von 704 bis 767 liegen, Nummer 0 und den anderen Nummer 1, dann müssen wir also eingeben:
70 POKE 2040,11:POKE 2041,13
Damit sind die Sprite-Zeiger gesetzt.

## Anschalten der Sprites

Wenn Sie jetzt freudig RUN (RETURN) eingetippt haben, um unsere Drudenfüsse zu sehen, werden Sie ein langes Gesicht gemacht haben: Nix zu sehen! Aber das ist wie bei einer Lampe: Sie haben den Lampenschirm, die Glühbirne, den Stecker eingesteckt und sie leuchtet nicht! Deswegen schalten wir sie jetzt ein, die Sprites. Der Schalter dafür sitzt im Sprite-Kontrollregister 53269 (siehe Folge 1 Tabelle der VIC-II-Chip-Register). Dort gibt es für jedes MOB ein Bit. Also Sprite Nummer 0 entspricht Bit Nummer 0 und so weiter. Ein Sprite ist eingeschaltet, wenn sein Bit auf 1 gesetzt ist. Wie man einzelne Bits setzt oder löscht, kennen wir noch aus der zweiten Folge. Spielen wir das hier nochmal an unserem Beispiel durch: Wir wollen Sprite Nummer 0 und Sprite Nummer 1 einschalten, müssen also die Bits 0 und 1 auf den Wert 1 setzen. Da gab es doch die OR-Funktion und eine sogenannte Maske:

XXXXXXXX irgendein binärer Inhalt von 53269
OR 00000011 Maske (= dezimal 3)
XXXXXX11 Ergebnis (Bits 0 und 1 sind = 1)

Das ergäbe die Programmzeile:
80 POKE 53269,PEEK(53269) OR 3

Will man einzelne MOBs mit der Nummer N einschalten, dann empfiehlt sich folgender Befehl:
POKE 53269,PEEK(53269) OR (2↑N)

| | Abschnitt 1 | Abschnitt 2 | Abschnitt 3 |
|---|---|---|---|
| Zeile 1 | Byte 0 | Byte 1 | Byte 2 |
| Zeile 2 | Byte 3 | Byte 4 | Byte 5 |
| . | . | . | . |
| . | . | . | . |
| . | . | . | . |
| Zeile 20 | Byte 57 | Byte 58 | Byte 59 |
| Zeile 21 | Byte 60 | Byte 61 | Byte 62 |

Bild 2. Die Sprite-Organisation im VIC-II-Chip

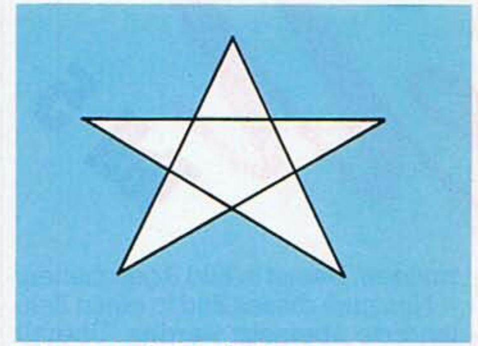


Bild 1. Ein Drudenfuß oder Pentagramm

Das Ausschalten geschieht durch die AND-Funktion. Jedes Bit, das mit 0 AND-verknüpft wird, wird dadurch gelöscht. Wenn wir also Sprite Nummer 1 ausschalten wollen, verwenden wir wieder eine Maske:

XXXXXX11 PEEK(53269), Sprite 0 und 1 eingeschaltet
AND 11111101 Maske (dezimal = 253)
XXXXXX01 Ergebnis: Bit 1 = 0, Sprite 1 ausgeschaltet, Bit 0 = 1, Sprite 0 eingeschaltet.

Allgemein kann man einzelne Sprites also abschalten mit POKE 53269, PEEK(53269) AND (255-2↑N), wobei wieder N die Sprite-Nummer ist.

Ach, Ihre Geduld wird schon auf eine harte Probe gestellt. Wenn Sie nämlich bis jetzt alle Programmzeilen brav eingegeben und gestartet haben, sehen Sie immer noch kein Sprite! Aber Sie müssen dem MOB noch sagen, wo er erscheinen soll!

## Schon wieder ein Koordinatensystem: Ort der Sprites

Vor den Erfolg haben auch die Commodore-Softwareplaner den Schweiß gesetzt! Denn nicht genug damit, daß wir den Bildschirm schon im Normalmodus in X-Richtung, in Y-Richtung in 25 Positionen und im Bit-Map-Modus in X-Richtung in 320, in Y-Richtung in 200 Positionen aufgeteilt finden, jetzt kommt noch eine Einteilung, die sogar noch über den sichtbaren Bildschirm hinausreicht! In Bild 4 sehen wir diese Aufteilung in 512 horizontale und 256 vertikale Koordinaten.

Für den Ort eines Sprites ist — wie in Bild angedeutet — die linke obere Ecke des Spritedefinitionsfeldes entscheidend. Also auch dann, wenn diese Ecke (wie in unserem Pentagramm) unsichtbar. So hat das im Bild gezeigte MOB die X-Koordinate 128 und die Y-Koordinate 120. Diese X- und Y-Werte muß man nun in die zur Sprite-Nummer

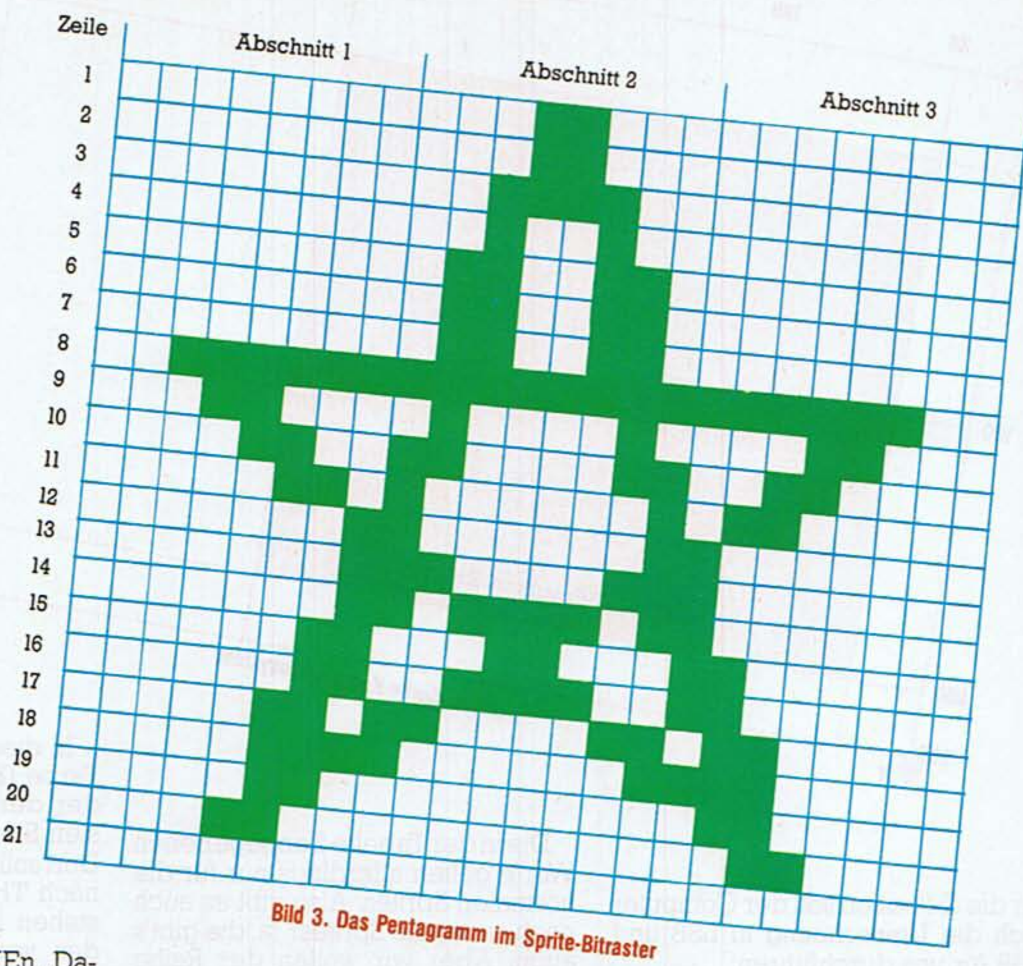



Bild 3. Das Pentagramm im Sprite-Bitraster

gehörigen Register einPOKEn. Dabei handelt es sich um folgende Speicherstellen:
X-Position von Sprite 0 : 53248
Y-Position von Sprite 0 : 53249

X-Position von Sprite 1 : 53250
Y-Position von Sprite 1 : 53251

und so weiter bis
X-Position von Sprite 7 : 53262
Y-Position von Sprite 7 : 53263

Um nun also endlich unser Sprite Nummer 0 sichtbar zu machen, geben wir ein:

```
100 POKE 53248,128:POKE 53249,120
```

So! Jetzt tippen Sie ein RUN (RETURN) und endlich: Da ist unser Pentagramm. Gefällt es Ihnen? Sprite 1, das zweite Pentagramm, soll am rechten Bildschirmrand auftauchen, also ungefähr bei X = 266 und bei Y = 130. Deswegen müßten wir aber in die entsprechende X-Positions-Speicherstelle für Sprite 1 eine Zahl größer als 255 einPOKEn! Wenn Sie's versuchen, meldet der Computer natürlich einen ILLEGAL QUANTITY ERROR. Geht also nicht! Anscheinend war das bisher noch nicht verzwickt genug. Jetzt müssen wir nämlich mal wieder das Hexadezimalsystem bemühen (die Sechzehnfingerlinge, erinnern Sie sich: Folge 2).

Dezimal 266 ist hexadezimal $010A. Jetzt zerteilen wir diese Zahl wieder in das LSB und das MSB:

```
$ 01    0A
MSB     LSB
=1      =0A
```

und rechnen wieder zurück ins Dezimalsystem:
MSB = 1
LSB = 10

Rechnen Sie mal nach: Das MSB kann auch bei der höchsten X-Position nie größer als 1 werden. Es gibt also nur zwei Fälle: Ist die X-Position größer als 256, dann ist das MSB 1, sonst ist es 0.

Deswegen speichert man die MSBs aller 8 Sprites in nur einem Byte. Ebenso wie beim Kontrollregister für das An- und Ausschalten gehört auch hier zu jedem Sprite ein Bit, also: Bit 1 gehört zu Sprite 1 und so weiter. Wenn nun also die X-Koordinate von Sprite 1 266 ist, dann ergibt die Aufspaltung, wie oben gezeigt, MSB = 1 und LSB = 10. Das Register für die MSBs ist Speicherstelle 53264. Die 10 (das LSB) wird also in die normale Speicherstelle für die X-Position gePOKEt. Das MSB ( = 1 ) ist also Bit 1 (weil Sprite 1) von Speicherstelle 53264. Hier ist also wieder eine OR-Operation nötig. Die Y-Position wird ganz normal eingegeben. Es ergibt sich also die Programmzeile:

```
110 POKE 53250,10:POKE 53264,
PEEK(53264)OR(2↑1):
POKE 53251,130
```

Wenn Sie jetzt einen Schwarzweiß-Monitor haben, sehen Sie nach RUN (RETURN) trotzdem nur unser Sprite Nummer 0. Farbmonitor-Eigner bewundern schon jetzt Sprite Nummer 1. Warum, das wird uns gleich noch beschäftigen.

Welche Koordinaten eines Sprites sind eigentlich möglich, und was sieht man dann? Um das zu erforschen, bauen wir uns ein Primitiv-Sprite:

```
20 FOR I = 0TO 62:POKE I+896,
255:NEXT I:POKE 2042,14:
   REM SPRITE NR. 2
```

Dann schalten wir ein:

```
90 POKE 53269,PEEK(53269) OR (↑2)
```

und bauen eine Abfrage ein für die Position:

```
130 INPUT"SPRITE 2:X,Y=";X,Y:IF
X=—1 THEN END
```

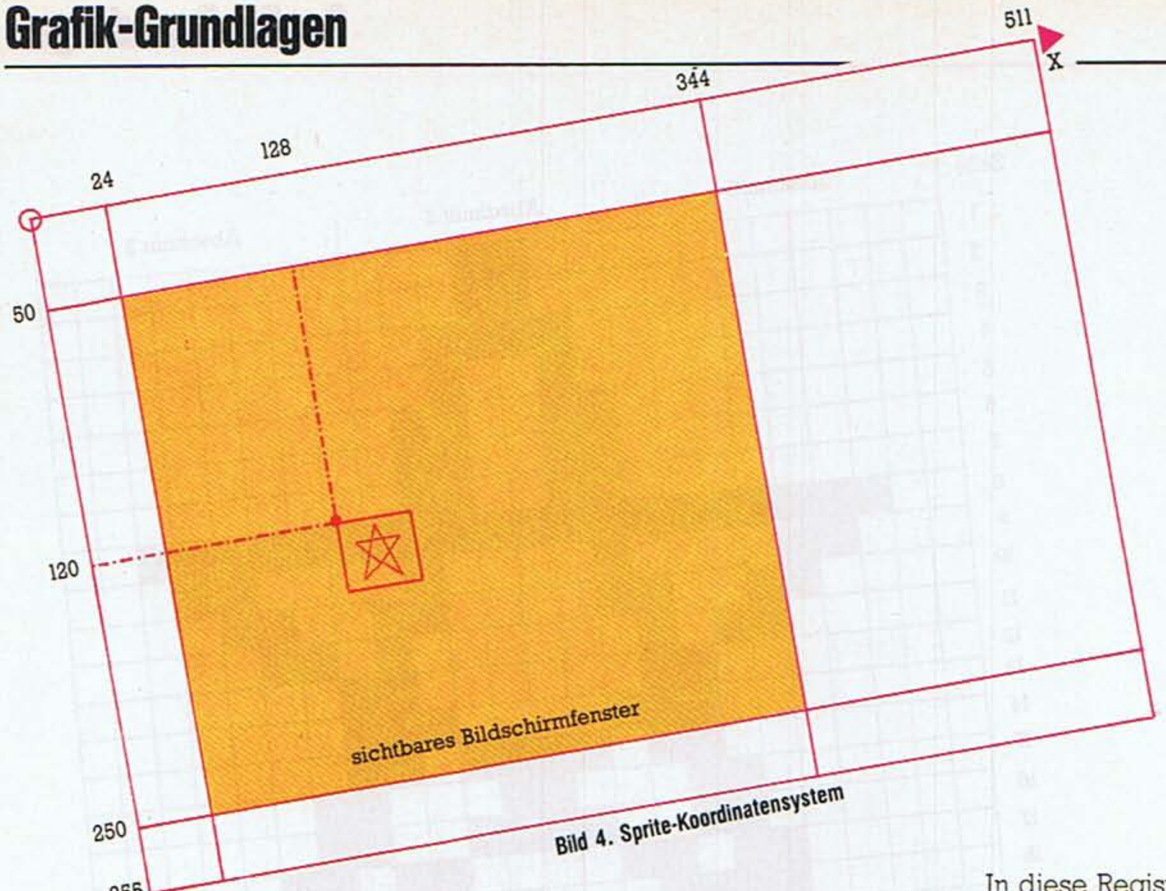



Bild 4. Sprite-Koordinatensystem

Für die X-Position soll der Computer noch die Umrechnung in LSB und MSB für uns durchführen:
140 X1=INT(X/256):X2=X—256*X1
Dann POKEn wir diese Koordinaten ein:
150 POKE 53252,X2:POKE 5253,Y: IF X1=1 THEN POKE 53264,PEEK (53264)OR(2↑2)

Wenn die X-Koordinaten kleiner als 256 sind, also das MSB = 0 ist, muß es natürlich gelöscht werden:
160 IF X1=0 THEN POKE 53264,PEEK(53264)AND(255—2↑27)

Schließlich lassen wir uns außer dem Sprite Nummer 2 auch noch die eingegebenen Koordinaten zeigen und kehren zur Abfrage zurück:
170PRINTCHR$(147),"x="x,"y="y: GOTO 130

Wenn Sie jetzt starten, können Sie alle möglichen Positionen für X und Y eingeben. Versuchen Sie doch mal X = 265, Y = 130. Da ist auch für die Schwarzweiß-Seher (ich bin auch einer) unser Sprite Nummer 1 zu erkennen. Warum, dazu kommen wir noch. Wenn Sie genug ausprobiert haben, dann geben Sie für X jetzt —1 und irgendeinen Y-Wert ein und das Programm ist beendet. Vermutlich haben auch Sie festgestellt, daß man unseren Testsprite ganz allmählich über den sichtbaren Bildschirmrand hinauswandern lassen kann. Es ergeben sich so die Grenzkoordinaten in Tabelle 2:

Die in der Tabelle 2 angegebenen Werte gelten allerdings nur für die normalen Sprites. Also gibt es auch noch anormale Sprites? Ja, die gibt's auch. Aber wir wollen der Reihe nach vorgehen. Wir können jetzt Sprites entwerfen, Sprite-Zeiger setzen, MOBs an- und wieder abschalten und sie an die richtige Stelle setzen.

## Mal wieder Farbe: Diesmal die von Sprites.

Eines haben wir bisher total vergessen: Welche Farbe soll unser Sprite haben und wie geben wir sie ihm? Auch hierfür gibt es natürlich wieder Register, die Sprite-Farben-Register und zwar die in der Tabelle 3.

```
READY.

10 :    REM**************************************************************
20 :    REM**                      S P R I T Y                         **
30 :    REM**                                                          **
40 :    REM**       MANFRED WALTER THOMA  2102 HAMBURG 93  ERNASTRASSE 10  **
50 :    REM**                                                          **
60 :    REM**************************************************************
70 POKE53280,0:POKE53281,0:POKE646,1:PRINTCHR$(147);
75 :    REM**************************************************************
80 :    REM**                   DATEN FUER DIE DREI SPRITES            **
90 :    REM**************************************************************
100 DATA0,0,0,0,0,0,0,0,0,0,0,0,14,0,0,17,0,0,32,128,0,64,64,0,155,32,0,128
110 DATA32,0,132,32,0,132,32,0,145,32,0,78,64,0,32,128,0,31,0,0,0,0,0,0,0,0
120 DATA0,0,0,0,0,0,0
130 DATA0,0,0,0,0,0,0,0,0,0,0,0,14,0,0,17,0,0,32,128,0,64,64,0,155,32,0,128
140 DATA32,0,132,32,0,132,32,0,128,32,0,78,64,0,64,64,0,32,128,0,31,0,0,0,0,0
150 DATA0,0,0,0,0,0,0,0
160 DATA0,0,0,0,0,0,0,0,0,0,0,0,14,0,0,17,0,0,32,128,0,64,64,0,155,32,0,128
170 DATA32,0,132,32,0,132,32,0,128,32,0,78,64,0,81,64,0,64,64,0,32,128,0,31,0
180 DATA0,0,0,0,0,0,0,0,0
```

In diese Register POKEt man die Farbe (Kennzahlen 0 bis 15) ein, in der der MOB erscheinen soll. Testen Sie mal: Ändern wir Zeile 130. Dort soll in der IF-THEN-Anweisung nach THEN anstelle von END jetzt stehen 180. Diese Zeile 180 schaltet das vorerst ausgediente Sprite 2 aus:
180 POKE 53269,PEEK(53269) AND (255—2↑2)

Damit eröffnen wir uns die Farbeingabemöglichkeit mit:
190 PRINT CHR$(147):INPUT"SPRITE 0,SPRITE 1 FARBEN";
F1,F2:IF F1=-1THEN210

Schließlich POKEn wir die Farben in die zu den Sprites gehörigen Register:
200 POKE 53287,F1:POKE53288, F2:GOTO190
210 END

Jetzt können Sie, bis Sie schließlich für F1 mal —1 eingeben, allerlei Farben durchprobieren. Wir erkennen nun auch, daß bislang Sprite 1 für

Schwarzweiß-Seher nicht erkennbar war, weil seine Farbe keinen Kontrast zur Hintergrundfarbe gebildet hat.

Wem's noch nicht bunt genug war bisher, der hat auch hier bei den Sprites die Möglichkeit, den Mehrfarben-Modus zu verwenden. Während im bisher gebrauchten Modus jedes Sprite-Definitions-Bit entweder 0 (= Hintergrundfarbe) oder 1 (Farbe des zum Sprite gehörigen Sprite-Farb-Registers) sein konnte, zählen — wie auch sonst im Mehrfarbenmodus — wieder Bit-Paare. Dabei stammt dann die jeweilige Farbe aus den in Tabelle 4 angegebenen Registern.

Unser Drudenfuß sieht somit dann aus wie in Bild 5 gezeigt.

Das wollen wir noch einmal auf dem Bildschirm ansehen. Wir schreiben ab Zeile 190 neu:

```
190 PRINT CHR$(147):INPUT"MOB1,
MOB2,MULTCOL1,MULTCOL2";F1,
F2,F3,F4:IFF1=—1THEN 220
200 POKE 53287,F1:POKE53288,F2:
POKE 53285,F3:POKE 53286,F4
220 END
```

So läuft natürlich noch nichts Neues, denn der Mehrfarben-Modus muß noch angeschaltet werden. Auch dazu gibt es wieder ein Register: 53276. Wie bei einigen anderen Sprite-Registern gehört auch wieder zu jedem MOB das entsprechende Bit, also zu Sprite 1 das Bit 1 und so weiter. Wenn dieses Bit auf 1 gesetzt ist, ist für das dazugehörige Sprite der Mehrfarb-Modus angeschaltet. Man kann sie also einzeln oder zusammen — ganz wie's beliebt — im Normal-Modus oder im Muticolor-Modus betrachten. Wir schalten Sprite 0 und Sprite 1 in den Mehrfarb-Modus mit Zeile

```
210 POKE 53276,PEEK(53276)OR3:
GOTO 190
```

Will man nur Sprite Nummer N umschalten, dann verwendet man wie gehabt: POKE 53276, PEEK(53276)OR(2↑N).

Das Zurückschalten in den Normalmodus geschieht dann durch Löschen der entsprechenden Bits: POKE 53276,PEEK(53276)AND (255—2↑N).

Auch hier habe ich das Programm so gebaut, daß man durch Eingabe von —1 und irgendwelchen drei anderen Zahlen aussteigen kann. Im Verlauf des Probierens werden Sie bestimmt gemerkt haben, daß man Sprites, die für den Mehrfarben-Modus gedacht sind, speziell konstruieren sollte unter Berücksichtigung der Bit-Paar-Zusammenstellungen. Unser Drudenfuß gefällt mir im Normal-Modus jedenfalls besser.

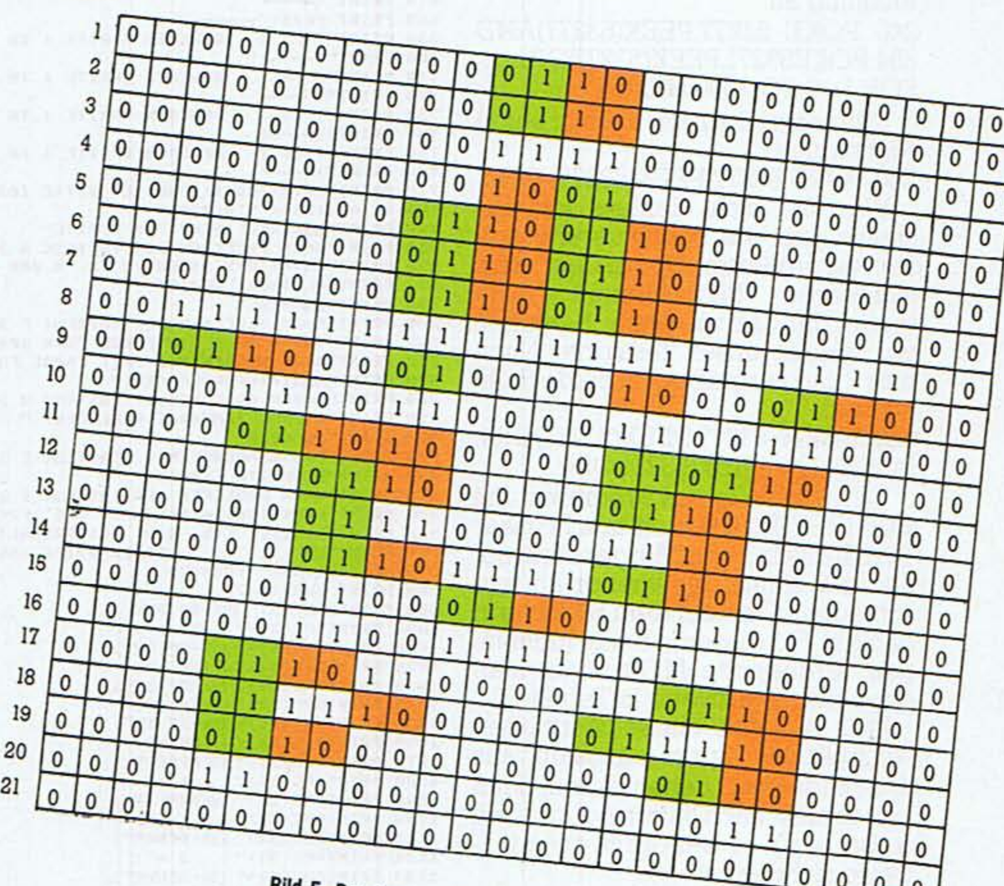



**Bild 5. Das Pentagramm im Modemfarbenmodus**

Deswegen schalten wir in Zeile 220 lieber den Mehrfarben-Modus aus:

```
220 POKE 53276,PEEK(53276) AND
252
```

## Anormale Sprites? Gequetschte und gezerrte MOBs

Die normalen Sprites bestehen aus 24 x 21 Bildpunkten und haben auf dem Bildschirm eine Ausdehnung von zirka drei Zeilen mal drei Spalten. Für einige Effekte ist es ganz nett, sie in ihrer Größe verändern zu können. Genau das ist möglich, und zwar in X-Richtung, in Y-Richtung oder in beide Richtungen gleichzeitig. Das Merkwürdige an diese Sache ist — außer dem manchmal recht verzerrten Aussehen —, daß die gleiche 24 mal 21 Pixel abgebildet werden, nur jedes Pixel ist vergrößert. Wenn sowohl in X- als auch in Y-Richtung vergrößert

```
190 :  REM*******************************************************
200 :  REM**              EINLESEN DER SPRITES UND TITELBILD      **
210 :  REM*******************************************************
220 PRINTTAB(12)"S  P  R  I  T  Y"
230 PRINTTAB(12)"================"
240 PRINT:PRINT:PRINT:PRINT
250 PRINTTAB(19)"BY"
260 PRINTTAB(12)"MANFRED W. THOMA"
270 PRINTTAB(12)"2102  HAMBURG 93"
280 PRINTTAB(12)" ERNASTRASSE 10"
290 FORI=13*64TO13*64+62:READX:POKEI,X:NEXT
300 POKE2040,13:                 REM** SPRITE 0 AUS BLOCK 13    **
310 FORI=14*64TO14*64+62:READX:POKEI,X:NEXT
320 POKE2041,14:                 REM** SPRITE 1 AUS BLOCK 14    **
330 FORI=15*64TO15*64+62:READX:POKEI,X:NEXT
340 POKE2042,15:                 REM** SPRITE 2 AUS BLOCK 15    **
350 V=53248:                     REM** STARTADRESSE DES VIC 2 CHIP**
360 POKE V+39,1:                 REM** FARBE FUER SPRITE 0      **
370 POKE V+40,1:                 REM** FARBE FUER SPRITE 1      **
380 POKE V+41,1:                 REM** FARBE FUER SPRITE 2      **
```

**Das Programm »Sprity«**

wurde, ist einfach jeder Bildpunkt viermal so groß wie vorher. Auch hier geschieht das natürlich wieder über Kontrollregister, von denen jedes Bit wieder zu einem Sprite gleicher Nummer gehört. Die Verdoppelung in X-Richtung wird durch eine 1 im zum Sprite gehörigen Bit des Registers 53277, die in Y-Richtung auf die gleiche Weise im Register 53271 geschaltet. Wenn wir also unser Sprite Nummer 0 in X-Richtung verdoppeln wollen, dann geben wir noch ein:

230 POKE 53277,PEEK(53277)OR1: FOR I=0TO2000:NEXT I

Dann sehen wir uns das nach einer kleinen Pause noch in Y-Richtung an:

240 POKE 53277,PEEK(53277)AND 254:POKE53271,PEEK(53271)OR1: FOR I=0 TO 2000:NEXT I

Jetzt vergrößern wir noch in beide Richtungen:

250 POKE 53277,PEEK(53277) OR 1

Wie Sie sich denken können, stimmt jetzt die Positionierung dieser vergrößerten Sprites auf dem sichtbaren Bildschirmteil nicht mehr. Um das zu testen, bemühen wir wieder unser Testsprite Nummer 2 und vergrößern in beide Richtungen:

120 POKE 53277,PEEK(53277)OR4: POKE53271,PEEK(53271)OR4

Wenn wir nun das Programm mit RUN (RETURN) starten, taucht unser Test-Sprite in vergrößterter Form auf und wir können es wieder an verschiedene Orte auf dem Bildschirm packen. Für dieses in beide Richtungen verdoppelte MOB findet man dann die Grenzwerte in Tabelle 5.

Will man also ein solchermaßen vergrößertes Sprite langsam aus dem Bildschirm ziehen lassen, dann ist das nicht nach links möglich, weil selbst bei X=0 der Sprite noch teilweise sichtbar ist.

Jetzt wissen wir eigentlich fast alles, was mit dem einzelnen Sprite zusammenhängt. Wenn Sie ein sich veränderndes Sprite darstellen wollen, so ist das zum Beispiel möglich, indem alle zu zeigenden Bewegungsphasen als Spritemuster im Speicher abgelegt werden und dann per Programm der Sprite-Zeiger auf den jeweils aktuellen Bewegungszustand umgeschaltet wird. So könnte man wie in Bild 6 die Abläufe A bis G im Speicher ablegen und zum Beispiel den Sprite-Zeiger für Sprite 3 zuerst auf das Muster A, dann nach entsprechender Verzögerung auf Muster B, dann C, D, E, F, G und schließlich wieder A richten. Wenn die einzelnen Spritemuster gut ge-

```
390 POKE V+23,7:                          REM** SPRITES 0-2 IN X-RICHTUNG  **
400 POKE V+29,7:                          REM** SPRITES 0-2 IN Y-RICHTUNG  **
410 POKEV,160:POKEV+1,160:                REM** X-Y-KOORDINATEN SPRITE 0   **
420 POKEV+2,160:POKEV+3,160:              REM** X-Y-KOORDINATEN SPRITE 1   **
430 POKEV+4,160:POKEV+5,160:              REM** X-Y-KOORDINATEN SPRITE 2   **
440 FOR I=1 TO10
450 POKEV+21,1:FOR II=1TO300:NEXT:        REM** EINSCHALTEN NUR SPRITE 0   **
460 POKEV+21,2:FOR II=1TO200:NEXT:        REM** EINSCHALTEN NUR SPRITE 1   **
470 POKEV+21,4:FOR II=1TO100:NEXT:        REM** EINSCHALTEN NUR SPRITE 2   **
480 NEXTI
490 POKEV+21,0:                           REM** AUSSCHALTEN ALLER SPRITES  **
500 PRINTCHR$(147)
510 DR$="|     ←▮▮▮▮▮▮▮"
520 POKE 650,128:                         REM**  REPEAT FUER ALLE TASTEN   **
530 : REM**************************************************************************
540 : REM**                       TASTENBELEGUNGSERKLAERUNG                     **
550 : REM**************************************************************************
560 PRINTCHR$(147);
570 PRINTTAB(9)"BEWEGUNGSSIMULATION"
580 PRINTTAB(9)"==================="
590 PRINT:PRINT"┌────┐"
600 PRINT"| F1 |   BEWEGEN SPRITE 0 IN +X RICHTUNG";
610 PRINT"├────┤
620 PRINT"| F3 |   BEWEGEN SPRITE 0 IN -X RICHTUNG";
630 PRINT"├────┤
640 PRINT"| F5 |   BEWEGEN SPRITE 0 IN +Y RICHTUNG";
650 PRINT"├────┤
660 PRINT"| F7 |   BEWEGEN SPRITE 0 IN -Y RICHTUNG";
670 PRINT"└────┘
680 PRINT:PRINT"┌────┐"
690 PRINT"| F2 |   BEWEGEN SPRITE 1 IN +X RICHTUNG";
700 PRINT"├────┤"
710 PRINT"| F4 |   BEWEGEN SPRITE 1 IN -X RICHTUNG";
720 PRINT"├────┤
730 PRINT"| F6 |   BEWEGEN SPRITE 1 IN +Y RICHTUNG";
740 PRINT"├────┤
750 PRINT"| F8 |   BEWEGEN SPRITE 1 IN -Y RICHTUNG";
760 PRINT"└────┘
770 PRINT:PRINTTAB(11)"WEITER BITTE TASTE"
780 GETA$:IFA$=""THEN780
790 PRINT"▯▯VERKLEINERN DER SPRITE"
800 PRINT"NACH DRUECKEN DER TASTE▯< V >■KANN MIT DEN";
810 PRINT"FUNKTIONSTASTEN IN +X,-X UND +Y,-Y  VER-";
820 PRINT"KLEINERT WERDEN."
830 PRINT"▯▯FARBEN
840 PRINT"NACH DRUECKEN DER TASTE▯< F >■KANN MIT DER";
850 PRINT"TASTE >F1< DIE FARBE FUER SPRITE0 GEAEN-";
860 PRINT"DERT WERDEN.MIT >F2< FARBE FUER SPRITE1.";
870 PRINT"▯▯HINTERGRUNDFARBE"
880 PRINT"DURCH DRUECKEN D. TASTE▯< H >■WIRD DIE"
890 PRINT"BILDSCHIRMFARBE GEAENDERT."
900 PRINT"▯▯ENDE"
910 PRINT"PROGRAMMENDE MIT  TASTE▯< E >■"
920 PRINT"▯▯SPRITE EIN/AUS"
930 PRINT"NACH DRUECKEN DER TASTE▯< O >■KANN MIT DEN";
940 PRINT"FUNKTIONSTASTEN >F1< UND >F2<SPRITE0 UND";
950 PRINT"SPRITE1 EIN- BZW. AUSGESCHALTET WERDEN"
960 PRINT"▯             ▯WEITER BITTE TASTE"
970 GETA$:IFA$=""THEN970
980 PRINT CHR$(147);
990 PRINTSPC(33)" ▯X-SP.0■";
1000 PRINTSPC(33)"|  50 ←";
1010 PRINTSPC(33)" ▯Y-SP.0■";
1020 PRINTSPC(33)"| 150 ←";
1030 PRINTSPC(33)" ▯X-SP.1■";
1040 PRINTSPC(33)"| 150 ←";
1050 PRINTSPC(33)" ▯Y-SP.1■";
1060 PRINTSPC(33)"| 150 ←";
1070 PRINTSPC(33)" ▯FARBE0■";
1080 PRINTSPC(33)"|   1 ←";
1090 PRINTSPC(33)" ▯FARBE1■";
1100 PRINTSPC(33)"|   1 ←";
1110 PRINTSPC(33)" ▯X-DEHN■";
1120 PRINTSPC(33)"|   3 ←";
1130 PRINTSPC(33)" ▯Y-DEHN■";
1140 PRINTSPC(33)"|   3 ←";
1150 PRINTSPC(33)" ▯SCHIRM■";
1160 PRINTSPC(33)"|   0 ←";
1170 PRINTSPC(33)" ───────";
1180 PRINTSPC(33)" ▯ON/OFF■";
1190 PRINTSPC(33)" ▯S0:■ ON";
1200 PRINTSPC(33)" ▯S1:■ ON";
1210 PRINT"▯               S P R I T Y            ";CHR$(19)
1220 POKEV,50:
1230 POKEV+2,150
1240 POKEV+21,3
1250 A=PEEK(V+30):                          REM** LESEN/LOESCHEN KOLLISONREG**
1260 : REM*************************************************************************
1270 : REM**                       TASTATUR ABFRAGE (GET)                      **
1280 : REM*************************************************************************
1290 GETA$:IFA$=""THEN1290
1300 IF A$="E"THENPRINTCHR$(147):POKEV+21,0:END
1310 IF A$="H"THEN 1730:                    REM** AENDERN DER BILDS.FARBE    **
1320 IF A$="V"THEN 1990:                    REM** SPRITES KLEINER/GROESSER   **
1330 IF A$="F"THEN 1830:                    REM** SPRITES FARBE              **
1340 IF A$="O"THEN 2210:                    REM** SPRITES EIN/AUS            **
1350 A=ASC(A$)-132:                         REM** UMRECHNEN FUNKTIONSTASTEN **
1360 IFA<0 OR A>8THEN1290:                  REM** KEINE FUNKTIONSTASTE       **
1370 ON A GOSUB  1490,1520,1550,1580,1610,1640,1670,1700
1380 : REM*************************************************************************
1390 : REM**                       KOLLISIONS ABFRAGE                          **
1400 : REM*************************************************************************
```

**Das Programm »Sprity« (Fortsetzung)**

```
1410 IF PEEK(V+30)=0 THEN 1290
1420 PRINT"S        !!! KOLLISION !!!"
1430 FOR I=0 TO 100:NEXT
1440 PRINT"S                          "
1450 GOTO 1290
1460 : REM*************************************************************
1470 : REM**                    SPRITE BEWEGEN X/Y-KOORD.            **
1480 : REM*************************************************************
1490 WE=PEEK(V):IFWE=255 THEN RETURN
1500 WE=WE+1:POKEV,WE
1510 Z=1:GOSUB2370:RETURN
1520 WE=PEEK(V):IFWE=1THEN RETURN
1530 WE=WE-1:POKEV,WE
1540 Z=1:GOSUB2370:RETURN
1550 WE=PEEK(V+1):IFWE=1THEN RETURN
1560 WE=WE-1:POKEV+1,WE
1570 Z=3:GOSUB2370:RETURN
1580 WE=PEEK(V+1):IFWE=198THEN RETURN
1590 WE=WE+1:POKEV+1,WE
1600 Z=3:GOSUB2370:RETURN
1610 WE=PEEK(V+2):IFWE=255 THEN RETURN
1620 WE=WE+1:POKEV+2,WE
1630 Z=5:GOSUB2370:RETURN
1640 WE=PEEK(V+2):IFWE=1THEN RETURN
1650 WE=WE-1:POKEV+2,WE
1660 Z=5:GOSUB2370:RETURN
1670 WE=PEEK(V+3):IFWE=1THEN RETURN
1680 WE=WE-1:POKEV+3,WE
1690 Z=7:GOSUB2370:RETURN
1700 WE=PEEK(V+3):IFWE=198THEN RETURN
1710 WE=WE+1:POKEV+3,WE
1720 Z=7:GOSUB2370:RETURN
1730 : REM*************************************************************
1740 : REM**                  HINTERGRUNDFARBE (BILDS.)              **
1750 : REM*************************************************************
1760 HF=PEEK(53280)
1770 IF HF=255 THEN POKE53280,0:POKE53281,0:WE=0:Z=17:GOSUB2370:GOTO 1290
1780 POKE53280,HF+1:POKE53281,HF+1
1790 WE=HF+1-240:Z=17:GOSUB2370
1800 GOTO 1290
1810 :
1820 :
1830 : REM*************************************************************
1840 : REM**                         SPRITE  FARBEN                  **
1850 : REM*************************************************************
1860 FOR I=1 TO50
1870 GETA$
1880 IFA$=CHR$(133) OR A$=CHR$(137)THEN 1910:REM** PRUEFEN  >F1<  ODER  >F2< **
1890 NEXT I
1900 GOTO 1290
1910 F1=PEEK(V+40):F2=PEEK(V+39):             REM** AKTUELLE FARBEN SPRITE 0+1**
1920 IF A$=CHR$(137) THEN 1960
1930 IF F2=255THEN POKEV+39,0:WE=0:Z=9:GOSUB2370:GOTO1290
1940 POKE V+39,F2+1:WE=F2+1-240:Z=9:GOSUB2370
1950 GOTO1290
1960 IF F1=255THEN POKEV+40,0:WE=0:Z=11:GOSUB2370:GOTO1290
1970 POKE V+40,F1+1:WE=F1+1-240:Z=11:GOSUB2370
1980 GOTO1290
1990 : REM*************************************************************
2000 : REM**                   VERKLEINERN DER SPRITES               **
2010 : REM*************************************************************
2020 FOR I=1 TO 100
2030 GETA$:IF A$=""THEN2060
2040 A=ASC(A$)-132
2050 IF A>0 AND A<9 THEN 2070
2060 NEXT I:GOTO 1290
2070 DX=PEEK(V+29):DY=PEEK(V+23)
2080 ON A GOTO 2100,2110,2120,2130,2140,2150,2160,2170
2090 GOTO1260
2100 DX=DX OR 1:GOTO2180
2110 DX=DX AND 254:GOTO2180
2120 DY=DY OR 1:GOTO2190
2130 DY=DY AND 254:GOTO2190
2140 DX=DX OR 2:GOTO2180
2150 DX=DX AND 253:GOTO2180
2160 DY=DY OR 2:GOTO2190
2170 DY=DY AND 253:GOTO2190
2180 POKE V+29,DX:WE=DX-4:Z=13:GOSUB 2370:GOTO1290
2190 POKE V+23,DY:WE-DY-4:Z=15:GOSUB 2370:GOTO1290
2200 GOTO1290
2210 : REM*************************************************************
2220 : REM**                   SPRITES EIN/AUS SCHALTEN              **
2230 : REM*************************************************************
2240 FOR I=1 TO 100
2250 GETA$:IF A$=""THEN2280
2260 IFA$=CHR$(133)THEN 2290
2270 IFA$=CHR$(137)THEN 2310
2280 NEXT I: GOTO 1290
2290 IF(PEEK(V+21)AND1)=1THENPOKEV+21,PEEK(V+21)-1:GOTO2330:REM*BIT 0 GESETZT ?*
2300 POKEV+21,PEEK(V+21)+1:GOTO2340
2310 IF(PEEK(V+21)AND2)=2THENPOKEV+21,PEEK(V+21)-2:GOTO2350:REM*BIT 1 GESETZT ?*
2320 POKEV+21,PEEK(V+21)+2:GOTO2360
2330 POKE211,37:POKE214,20:SYS58640:PRINT"OFF":GOTO1290
2340 POKE211,37:POKE214,20:SYS58640:PRINT" ON":GOTO1290
2350 POKE211,37:POKE214,21:SYS58640:PRINT"OFF":GOTO1290
2360 POKE211,37:POKE214,21:SYS58640:PRINT" ON":GOTO1290
2370 :REM*************************************************************
2380 :REM**                       CURSOR SETZEN (X-Y)                **
2390 :REM*************************************************************
2400 POKE211,33:POKE214,Z:SYS58640:PRINTLEFT$(DR$,9+LEN(STR$(WE)));WE:RETURN
2410 :
READY.
```

**Das Programm »Sprity« (Schluß)**

macht und die Verzögerungsschleifen richtig abgestimmt sind, kann so eine Art Zeichentrickfilm ablaufen, der nun auch noch durch die anderen bisher gelernten Sprite-Eigenheiten (Vergrößern, Position, Farbe etc.) veränderbar ist. Wie Sie unschwer erkennen, sind Ihrer Phantasie keine Grenzen gesetzt, und ich würde mich freuen, von Ihnen mal so einen witzigen Trickablauf sehen zu können. Auf einem ähnlichen Prinzip basiert auch ein Programm von Hans Grigat in Happy-Computer, Ausgabe Nummer 11 (1983), Seite 99 ff. Überhaupt lohnt es sich, sich dieses Programm mal genau anzusehen, weil hier die anfangs erwähnte Möglichkeit der Sprite-Daten-Verschiebung genutzt wurde.

## Wer hat Vorfahrt? Prioritäten

Wir wollen uns jetzt noch um die Beziehung von Sprites zu ihrer Umwelt (also zu anderen Sprites und/oder zu Zeichen auf dem Bildschirm) kümmern. Sie erinnern sich vielleicht an unseren Versuch, unser kleines Test-Sprite über den Bildschirm zu bewegen und an ein Ergebnis davon, nämlich, daß auch Schwarzweiß-Sehern plötzlich bei Eingabe der abgedruckten Bildschirmposition das Sprite Nummer 1 sichtbar war. Wenn also — wie in diesem Fall — zwei Sprites sich überdecken, welches von beiden wird dann gezeigt? Siehe dazu das Bild 7, auf dem die Situation abgebildet ist. Wir sehen da: an den Stellen, wo Sprite 1 Bits mit dem Wert 0 vorliegen hat, ist Sprite 2 zu sehen. Wo aber der Bitwert des Sprite 1 gleich 1 ist, wird Sprite 2 durch Sprite 1 verdeckt. Das MOB 1 hat eine höhere Priorität als MOB 2. Der VIC-II-Chip organisiert die Prioritäten von Sprites untereinander also in folgender Weise: Höchste Priorität hat Sprite 0, dann folgt Sprite 1, Sprite 2 und so weiter bis zum Schlußlicht Sprite 7.

Wenn Sie also Programme planen, in denen Sprites aneinander vorbeiziehen sollen, denken Sie daran, daß an diese Vorfahrtsregelung nichts zu ändern ist. Unter Umständen muß man dann die Sprite-Zeiger umwechseln, um die Prioritäten umzukehren: Man macht dann beispielsweise für den Augenblick der Überschneidung aus Sprite 7 zum Beispiel Sprite 1 und umgekehrt.

Eine andere Vorfahrtsregelung gilt, wenn Sprites und Bildschirmzei-

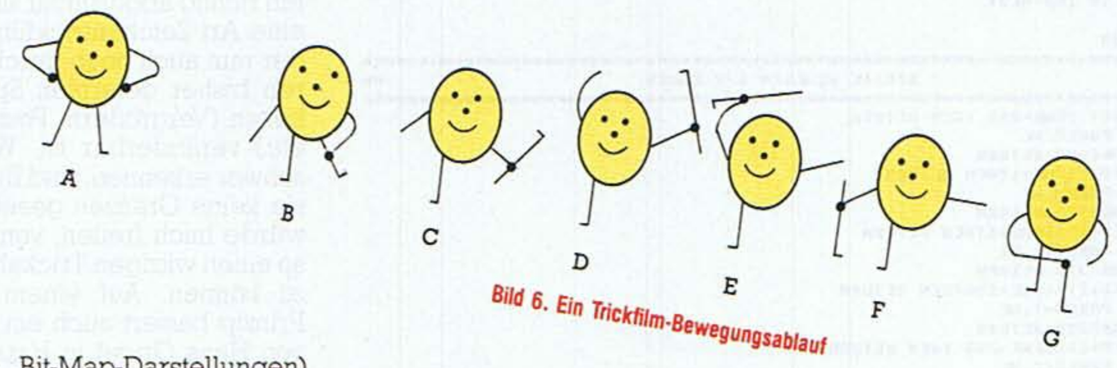


Bild 6. Ein Trickfilm-Bewegungsablauf

len (oder Bit-Map-Darstellungen) aufeinandertreffen. Hier haben wir ein Kontrollregister zur Hand (mal wieder eines!), Speicherstelle 53275, wo wieder jedem Sprite ein Bit entspricht (also Bit 0 entspricht Sprite 0 und so weiter). Wenn nun dieses Bit den Wert 0 hat, steht das dazugehörige MOB vor den Bildschirmdarstellungen, andernfalls verkrümelt es sich dahinter.

Das wollen wir uns mal ansehen! Starten Sie das Programm nochmal und wenn Sie an die Farbabfrage kommen, geben Sie dem Sprite 0 die Farbe 0 (= schwarz). Beenden Sie das Programm in der angegebenen Weise und wenn sich READY gemeldet hat. LISTen Sie das ganze. Jetzt steht unser schwarzes Pentagramm vor dem Listing. Geben sie nun im Direktmodus ein:
POKE 53275,1 (RETURN). Siehe da: Sprite 0 versteckt sich hinter dem Listing.

Wir können also zusammenfassen: Überall dort, wo auf dem Bildschirm (durch Buchstaben oder andere Zeichen) ein Bit gesetzt ist, verschwindet dahinter ein gegebenenfalls vorhandener Sprite-Bildpunkt (wenn natürlich das zum Sprite gehörige Bit im Register 53275 gesetzt ist). Nur dort, wo auf dem Bildschirm ein Bit nicht gesetzt ist (also 0 ist), sieht man einen dort vorhandenen Sprite-Bildpunkt.

Etwas komplizierter liegen die Verhältnisse, wenn wir das MOB im Mehrfarben-Modus vorliegen haben. Hier werden nämlich auch Bildschirmbitpaare 01 (von Zeichen oder Bit-Map-Darstellungen) genauso behandelt wie Bitpaare 00. Das heißt, auch an solchen Stellen wird ein eventuell vorhandenes Sprite-Bitpaar dargestellt.

## Zusammenstöße: Kollisionsregister

Erfreulicherweise haben die Software-Planer des C 64 auch zwei Möglichkeiten vorgesehen, Zusammenstöße in Registern abzufragen. Weil es zwei Sorten von Zusammenstößen gibt (Sprite kollidiert mit Sprite und Sprite kollidiert mit Zeichen), kann man zwei Register abfragen.

Da hätten wir zunächst das Sprite-Sprite-Kollisions-Register 53278. Auch hier gehört zu jedem Bit ein Sprite, wie bei den anderen Kontroll-Registern. Von einem Zusammenstoß spricht man in Sprite-Kreisen

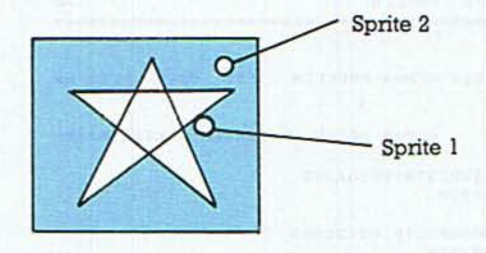


Bild 7. Sprite 1 überdeckt Sprite 2

immer dann, wenn undurchsichtige Teile der MOBs aufeinandertreffen. Überlappen sich nur die Teile, deren Bits bei der Sprite-Definition auf 0 gesetzt wurden, so zählt das nicht als Kollision. Die Bits der Sprites, die in den Zusammenstoß verwickelt sind, werden auf 1 gesetzt, zum Beispiel erzeugt ein Zusammenstoß von Sprite 0 mit Sprite 1 folgenden Inhalt des Registers 53278: 00000011 = dezimal 3.

Wenn in einem Supercrash acht Sprites kollidieren, steht im Sprite-Sprite-Kollisionsregister 255. Übrigens werden auch Zusammenstöße registriert, die außerhalb des sichtbaren Bildschirms liegen (siehe Bild 4).

Wenn ein Sprite mit Bildschirmdarstellungen (Zeichen und so weiter) zusammenstößt, wird in Register 53279 das zu ihm gehörende Bit gesetzt. Stößt also Sprite Nummer 1 mit zum Beispiel einem A zusammen, dann liest man aus dem Register 53279: 00000010 = dezimal 2. Apropos lesen: Die Register 53278 und 53279 sind so gestaltet, daß sie nach dem Herauslesen (PEEKen) gelöscht sind! Deswegen empfiehlt es sich, wenn man diese Inhalte danach noch braucht, sie in Variablen abzulegen. Auch bei dieser Sorte Zusammenstößen zählen nur diejenigen, bei denen undurchsichtige Sprite-Teile kollidieren. Wenn übrigens Bildschirminhalte horizontal aus dem sichtbaren Bereich »herausgescrollt« worden sind (zum »Scrollen« kommen wir in der nächsten Folge), und dabei ein Zusammenstoß mit einem Sprite geschieht, wird ebenfalls ein Bit in 53279 gesetzt.

Manfred Thoma hat das anliegende Programm »Sprity« geschrieben, mit dem Sie einige Sprite-Eigenschaften und die dazugehörigen Registerveränderungen beobachtet werden können.

Obwohl es zu den MOBs noch einiges zu sagen gäbe (wie kommen sie auf den Bildschirm, schnelles Steuern und so weiter), soll das Thema hiermit abgeschlossen sein. Ich denke, daß wir in dieser Folge die kleinen Kobolde schon weitgehend entzaubern konnten mit Hilfe unserer Pentagramme. Zu den Sprites gibt es mehr Literatur als zu vielen anderen C 64-Themen. Hier eine kleine Auswahl:

— Zwei Artikel wurden schon genannt (von H. Grigat und H. Kunz)
— Herbert Kunz hat auch schon etwas über schnelles Bewegen von MOBs geschrieben im 64'er, Ausgabe 4/83 auf Seite 70 f.
— Einen Überblick vor allem über die Anwendung von Sprites in Spielen geben Schneider und Ebert in den Bänden 1 und 3 des Commodore 64-Buches. Erschienen 1984 in Haar bei München im Markt & Technik Verlag. Diese beiden Bände sind auch von den sehr gut überschaubaren Programmbeispielen her zu empfehlen.

(Heimo Ponnath/aa)

# 4. Alle Tasten-, Zeichen- und Steuercodes

## Teil und Schluß

**In dem letzten Teil dieses Kurses finden Sie eine Zusammenfassung der Methoden, wie man in Basic Tasten abfragen kann.**

Wir haben insgesamt vier Methoden kennengelernt und verwendet, um das Drücken einer Taste im Programm abzufragen:

**1. Tastencode in Speicherzellen 203/653**

```
10 A=PEEK(203)
20 B=PEEK(653)
30 IF A = ZEICHENCODE AND
   B=STEUERCODE THEN
   AKTION
```

**2. ASCII-Code im Tastaturpuffer**

```
100 POKE 198,0
110 A=PEEK(631)
120 IF A=ASCII-CODE THEN AK-
    TION
```

**3. Abfrage des ASCII-Codes mit GET/INPUT**

```
200 GET A$
210 IF A$<>CHR$(ASCII-CODE)
    THEN AKTION 1
220 AKTION 2
```

**4. Abfrage des Gänsefuß-Modus mit GET/INPUT**

```
300 GET A$
310 IF A$ »ZEICHEN« THEN AK-
    TION 1
320 AKTION 2
```

Diese vier Methoden haben alle eins gemeinsam:

Sie können immer nur eine einzelne Taste abfragen. Zwei oder gar mehrere Tasten gleichzeitig oder kurz hintereinander gedrückt ergeben keine sinnvollen Resultate.

Jetzt möchte ich Sie an meine allererste Darstellung im Aprilheft des »64'er«, Seite 115, erinnern, nämlich wie die Tasten elektrisch angeordnet sind und wie das Betriebssystem sie abfragt. Ich möchte das hier noch einmal darstellen, erstens weil es eine gute Überleitung bildet zu meiner Methode der Vielfach-Tastenabfrage, zweitens weil meine damalige Darstellung nicht vollständig war.

In Bild 1 ist noch einmal die VC 20-Tastenanordnung dargestellt, in Bild 2 diejenige des C 64. Trotz des Unterschiedes der elektrischen Anordnung ist bei beiden Computern die Abfragemethode identisch, nur die dafür benötigten Register haben unterschiedliche Adressen.
Zur Erinnerung:

Das Betriebssystem wählt der Reihe nach die senkrechten Spalten dadurch an, daß es die Zahl, die am Fuße jeder Spalte steht, in das Spalten-Register 37152 (beziehungsweise 56320) schreibt. Diese Zahl ergibt in dualer Darstellung eine 0 an dieser Stelle.

Für jede Taste, die in der angewählten Spalte gedrückt ist, wird eine 0 in das andere Register 37153 (56321) geschrieben. Diese Dualzahl ergibt einen Dezimalwert, welcher aus dem Register herausPEEKbar ist.

Das, was das Betriebssystem macht, machen wir ihm nach, zuerst für den VC 20:

```
10 POKE 37152,247
20 PRINT PEEK (37152);PEEK(37153)
30 GOTO 10
```

und für den C 64:

```
10 POKE 56320,127
20 PRINT PEEK(56320);PEEK(56321)
30 GOTO 10
```

In Zeile 10 ist noch ein weiterer Unterschied zwischen den beiden Computern zu sehen. Beim VC 20 habe ich die Spaltenzahl 247 gewählt, weil in dieser Spalte die STOP-Taste liegt. Diese Spalte ist nämlich, wie wir in Teil 1 ja schon festgestellt haben, die einzige Spalte, die wir mit Basic abfragen können. Bei POKEn der anderen sieben Spaltenzahlen in Zeile 10 wirft uns die 60mal in der Sekunde stattfindende Überprüfung der STOP-Taste aus dem Programm.

Beim C 64 ist das die Spalte 127, wie es uns ein Blick auf das Bild 2 zeigt.

Ich freue mich übrigens, daß ein aufmerksamer Leser aus Wien diesen Unterschied sofort bemerkt und mich darauf aufmerksam gemacht hat. Aber ich hatte damals nur einen VC 20 zur Verfügung, und mangels eigener Erprobung ist es mir nicht aufgefallen. Das hat sich übrigens jetzt geändert.

Diese Tastenabfrage hat den großen Vorteil, daß mehrere Tasten gleichzeitig drück- und abfragbar sind. Jede gedrückte Taste erzeugt eine 0 im »Reihen-Register«. Mit dem kleinen Programm oben können Sie es ausprobieren. Drücken Sie alle Tasten der Spalte 247 (127), die STOP-Taste bitte als letzte! Der rechte Zahlenstreifen auf dem Bildschirm zeigt die 0.

Lassen Sie die STOP-Taste (die 1-Taste) los, und es erscheint die 1, bei Loslassen der linken SHIFT-Taste (der ←-Taste) zusätzlich erhalten wir die 3. Das ist die Dezimalzahl für 00000011, die beiden Einser entstehen durch die losgelassenen Tasten.

Jede mögliche Tastenkombination in einer Spalte hat ihre spezielle und abfragbare (!) Codezahl im Register 37153 (56321), insgesamt 256 Kombinationen. Ist das nichts?

Diese Methode der Mehrfach-Tastenabfrage haben wir bereits in der Ausgabe 4/84 im ersten Teil meines Aufsatzes erfolgreich eingesetzt.

Heute möchte ich diese Methode auf alle acht Spalten, also auf alle Tasten erweitern. Dazu müssen wir das oben erwähnte Hindernis, nämlich den Rausschmiß durch das Betriebssytem, überwinden. Dazu gibt es zwei Methoden. Methode 1 will ich nur kurz erwähnen — sie ist einen eigenen Aufsatz wert.

Man kann durch Beeinflussung der Speicherzellen 788 und 789 die Pause (Interrupt) zur Tastenabfrage des Betriebssytems künstlich verlängern und sie zur eigenen Abfrage benutzen.

Die zweite Methode ist einfacher. Wir schreiben das Programm oben (Zeilen 10 und 20) in Maschinensprache. Das läuft dann so schnell ab, daß es innerhalb zweier Unterbrechungen ausgeführt und somit vom Betriebssystem nicht gestört wird.

Da ich nicht annehme, daß alle Leser dieses Kurses in Maschinencode programmieren können, zeige ich es Ihnen einfach als Kochrezept. Es besteht aus einer Reihe von Zahlen, die über READ ... DATA in vorbestimmte Speicherplätze gePOKEt und von dort dann mit SYS gestartet werden.

Für den VC 20 gilt:

```
100 DATA 169, [254] ,141,32,145,173,
33,145,141,255,29,96
```

Für den C 64 gilt:

```
100 DATA 169, [254] ,141,0,220,173,1,
220,141,255,29,96
```

Ich habe einige Zahlen gekennzeichnet:

— Die zweite Zahl der DATA-Reihe in den Kästchen ist die Zahl der Spalte, die angewählt wird (ich habe 254 gewählt).

— Die 32 und 145 (0,220 beim C 64) ergeben die Registeradresse 37152 (56320), die DATA-Werte 33 und 145 (1,220) die Registeradresse 37153 (56321), die Werte 255,29 ergeben eine Speicherzelle 7679, auf die ich noch zurückkomme.

Die Registeradressen sind, wie immer bei Commodores 8-Bit-Computern, mit zwei Zahlen, das heißt mit zwei Bytes dargestellt.

---

REGEL:
LSB + 256*MSB = Adresse
32 + 256* 145 = 37152
1 + 256* 220 = 56321
LSB = niederwertiges Byte
MSB = höherwertiges Byte

---

Diese 12 DATA-Zahlen, ich nenne sie X, werden eingelesen mit:

```
110 FOR K=1 TO 12
120 READ X
140 NEXT K
```

Damit diese Zahlen vom Basicprogramm nicht überschrieben oder gelöscht werden (was dasselbe ist), POKEn wir sie an das Ende des VC 20-Speichers ohne Erweiterung, nämlich ab Speicherzelle 7661. Für den C 64 geht es mit denselben Adressen natürlich allemal.

```
130 POKE 7660+K,X
```

Um ganz sicher zu gehen, daß mit dem Maschinenprogramm nichts passiert, schützen wir es, indem wir dem Computer vorgaukeln, daß sein für Basic zur Verfügung stehender Speicher bei Adresse 7659 aufhört. Diese Speichergrenze steht in den Speicherzellen 51/52 und 55/56, und sie kann natürlich durch POKEn anderer Zahlen willkürlich verändert werden.

Die entsprechenden Werte für die Grenze 7659 sind 235 und 29. Machen wir die Probe:

PRINT 235+256*29 (RETURN)
ergibt 7659.

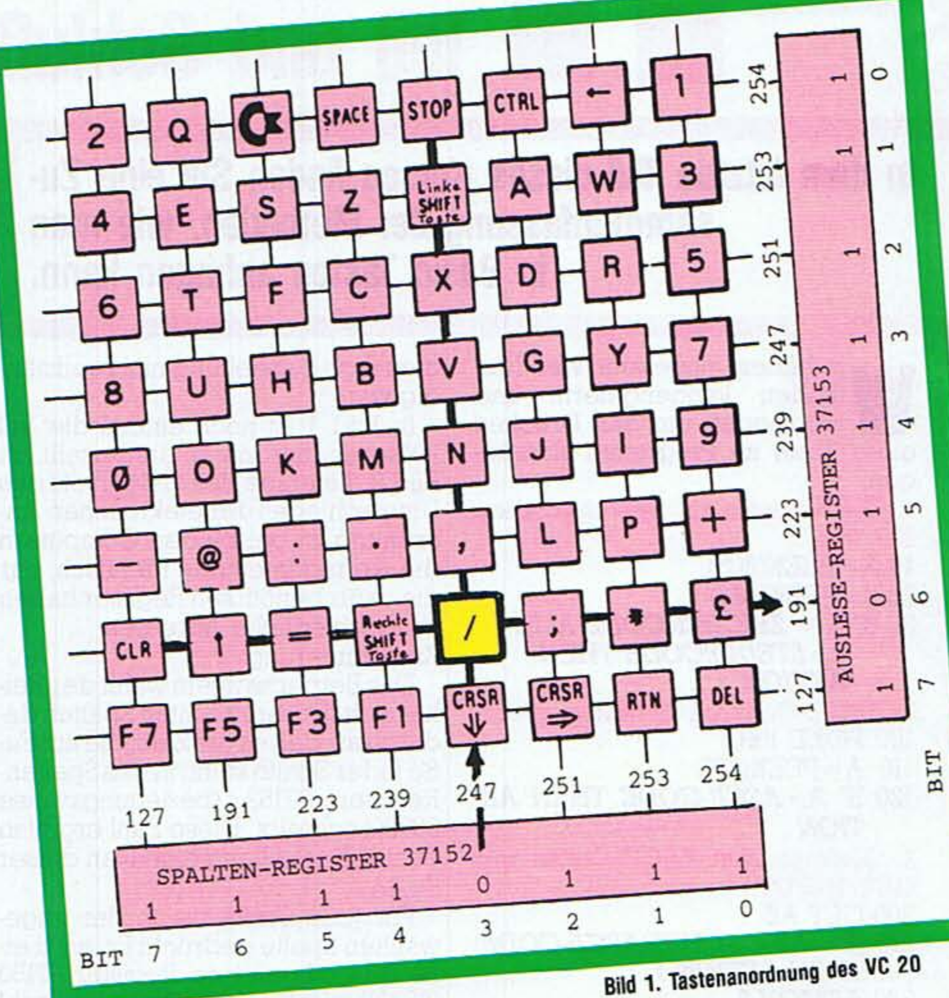



Bild 1. Tastenanordnung des VC 20

```
20 POKE 51,235
30 POKE 52,29
40 POKE 55,235
50 POKE 56,29
```

Falls Sie diese Speicherzellen und ihre Anwendung nicht kennen, bitte ich um Zuschrift, da ich es hier aus Platzgründen nicht näher erläutern kann.

So, nun ist das Maschinenprogramm gespeichert und gesichert. Wir starten das Programm mit SYS-Aufruf der 1. Adresse und PEEKen danach in das Register 37153 (56321), genau wie vorher.

```
150 SYS 7661
160 PRINT PEEK(37153): REM VC 20
160 PRINT PEEK(56321): REM C 64
170 GOTO 150
```

Nach RUN erzeugt der Rücksprung den schon obligatorischen Zahlenstreifen mit 255 für »keine Taste gedrückt«. Wenn wir die Taste 3 (RETURN beim C 64) drücken, dann muß entsprechend Bild 1 und 2 (Spalte 254) die Zahl 253 erscheinen. Das tut sie auch, aber immer wieder unterbrochen durch 255. Das kommt daher, daß das Auslesen des Registers 37153 (56321) in Basic doch schon zu langsam ist. Wir müssen daher die Funktion der Zeile 160 ebenfalls in das Maschinenprogramm einbauen, was ich in weiser Voraussicht in Zeile 100 schon vorbereitet habe. Der Trick ist nämlich dabei, daß der Inhalt des Registers vom Maschinenprogramm in eine vor dem bösen Basic geschützte Speicheradresse gebracht wird. Da wir ja einen gesicherten Bereich bereits haben, habe ich darauf die Zelle 7679 ausgewählt.

Bitte ändern Sie die Zeile 160 ab:

```
160 PRINT PEEK(7679)
```

Jetzt haben wir's geschafft.

Natürlich können wir mit diesem Programm alle acht Spalten abfragen. Wir brauchen bloß in Zeile 100 die Zahl, die hier im Kästchen steht, abändern, zum Beispiel in 253, und schon reagiert das Programm auf alle Tasten dieser Spalte.

Ich habe mir den Luxus erlaubt

und unser Programm von oben erweitert, indem ich der Reihe nach alle acht Spalten aufrufe (siehe Listing 1). Dazu habe ich in Zeile 100 die Spaltennummer auf 0 gesetzt, was eigentlich unnötig aber einleuchtend ist, POKE aber dann jeweils die Spaltenzahl (in den Zeilen 200, 300, 400 etc.) auf diesen Platz, der die Adresse 7662 hat.

Das Programm ist immer noch schnell genug, um innerhalb der zur Verfügung stehenden Zeit von 1/60 Sekunde alle Abfragen durchzuführen.

Ein Appell an alle 64er: Wenn Sie die Semikolons in den Zeilen 220, 320, 420 etc. (außer 920!!!) richtig plaziert haben, erscheinen die Zahlen für alle acht Spalten brav nebeneinander.

Beim VC 20 haben wir ein kleines Problem, hervorgerufen durch die kleinere Zeilenlänge, die uns eine Darstellung aller acht Zahlen nebeneinander nicht erlaubt.

Ich schlage deshalb vor, nach vier Zahlen ab Zeile 520 die Reihenfolge zu unterbrechen (Semikolon weglassen) und an den Anfang (Zeile 200) zurückzuspringen, es sei denn, die f-1-Taste wurde gedrückt (Zeile 560). Dann nämlich kommen die nächsten vier Zahlen dran — sogar in einer anderen Farbe. Eine weitere Tastenabfrage in Zeile 960, diesmal der .....Taste — ja schauen Sie halt in der ASCII-Tabelle nach, welche Taste dieses Zeichen hat (Methode des waagrechten Sprunges zwei Spalten weiter) — entscheidet zwischen Wiederholung der zweiten Gruppe oder Rücksprung auf die ersten vier Zahlen. Wie gesagt, die Verwendung der Semikolons an der richtigen Stelle ist wichtig.

Jetzt ist die Gelegenheit da zum Experimentieren:
— mehrere Tasten aus verschiedenen Spalten
— mehrere Tasten aus derselben Spalte

Ja, und irgendwann bleibt Ihnen das Programm mit BREAK IN .... stehen!!

Leider ist das schon wieder der STOP-Tasten-Effekt, den wir nicht ganz loswerden. Wenn nämlich Tasten der waagrechten Reihe 254 (beziehungsweise 127) gedrückt werden, interpretiert dies das Betriebssystem als STOP-Taste, die ja in dieser Reihe liegt, und stoppt das Programm.

**Bild 2. Tastenanordnung des C 64**

Uns soll das aber nicht bedrücken, denn wir können in unserem Programm den Gebrauch dieser Tastenreihe einfach vermeiden. Für unsere Zwecke reichen alle andern Tasten allemal aus. Man muß diese Einschränkung nur berücksichtigen.

Zusammenfassend sei gesagt, daß Sie in einem Programm jetzt alle Kombinationen aller Tasten abfragen können mit
IF PEEK(7679) = ..... THEN .....

Sie müssen sich lediglich die entstehenden Dualzahlen im Register 37153 (56321) in Dezimalzahlen umrechnen oder sie vorher ausprobieren.

Ich will auch am Schluß meiner Darstellung der Gewohnheit treu bleiben und das Kochrezept in einem kleinen Gericht anwenden (Listing 2).

## Ein Spiel für vier Personen

Um mir und Ihnen die Sache leichter zu machen, verwende ich möglichst viele Teile aus den vorherigen Programmen.

**Spielidee:**
— Jeder Spieler wählt eine von vier bestimmten Tasten.
— Auf dem Bildschirm wird ein beliebiger Buchstabe »angesagt«
— Der Bildschirm füllt sich langsam mit Zufalls-Buchstaben
— Sobald der »angesagte« Buchstabe erscheint, sollen die Spieler ihre Taste drücken
— Wer zuerst drückt, hat gewonnen!

Programmbeschreibung:
Zeile 10 bis 50
— schützt das Maschinenprogramm (wie vorher)
Zeile 100 bis 140
— liest das Kochrezept »Maschinenprogramm« ein
Zeile 215 und Zeile 530
— stellen einen besonderen Trick dar. Man nennt das eine »Flagge setzen« (set a flag). Die Abfrage der Tasten soll nämlich nur dann funktionieren, wenn ein Buchstabe (später VB genannt) mit dem anfangs »angesagten« Buchstaben BU übereinstimmt. Wenn VB = BU dann wird in eine »sichere« Speicherzelle (ich habe 830 gewählt) eine Zahl (hier 250) hineingePOKEt. Diese Zahl in 830 wird bei der Abfrage ebenfalls abgefragt. Bei Beginn des Spieles muß diese Flagge natürlich eingeholt, das heißt auf 0 gesetzt werden (Zeile 215).
Zeile 220 bis 270

— gibt Anweisungen und teilt den Spielern ihre Taste zu (von mir willkürlich ausgewählt)
Zeile 280/300
— löscht den Bildschirm und färbt alles schwarz
Zeile 310/320
— gibt Anweisungen
Zeile 330
— wählt per Zufallsgenerator (wie vorher) einen Buchstaben BU aus, zwischen 1 und 10 (A bis J in Bildschirm-Code!)
Zeile 340/350
— druckt diesen Buchstaben BU genau an das Ende des Textes der Zeile 320 (Bildschirmspeicher-Platz und Farbspeicher-Platz ausrechnen!)
Zeile 260/270 und 370/380
— schalten den Text weiter mit irgendeiner Taste

Zeile 400
— verzögert das Erscheinen des 1. Buchstabens
Zeile 520
— wählt wieder per Zufallsgenerator einen Buchstaben, diesmal VB, aus (zwischen A und J)
Zeile 540
— wählt per Zufallsgenerator einen Platz C auf dem Bildschirm aus, auf den der Buchstabe VB gePOKEt wird

Beim VC 20 fülle ich den ganzen Bildschirm von Platz 0 bis 505. Beim C 64 wäre das der Bereich von 0 bis 1000. Damit aber den Spielern wegen der viel kleineren Buchstaben des C 64 die Augen nicht übergehen, habe ich die eigentliche Formel C = INT(RND(0)*1000) in Zeile 540 so abgeändert, daß die Buchstaben nur auf jeden 2. Platz gePOKEt werden können.
Zeile 550/560
— die Buchstaben VB werden in weiß gePOKEt
Zeile 610
— leiert die Tastenabfrage 15 mal hintereinander durch. Diese Zahl bestimmt auch die Geschwindigkeit, mit der die Buchstaben auf dem Bildschirm erscheinen. Ihre Verkleinerung erhöht die Schwierigkeit des Spieles
Zeile 620 bis 740
— fragt die in Zeile 220 bis 250 definierten Tasten und die Flagge in 830 ab. Beim VC 20 ist jede der gewählten Tasten in einer anderen Spalte, daher sind vier POKEs in 7662 notwendig. Beim C 64 liegen je zwei Tasten in einer Spalte, daher nur zwei POKEs in 7662. Da aber gleichzeitige Tastendrücke vorkommen können (in einer Spalte natürlich), fragen Zeile 660 und 710 diesen speziellen Fall ab.
Zeile 810 bis 840 beziehungsweise 850
— meldet den Sieger (Aussprung aus der Abfrage)

## Zusammenfassung und Schluß

Diese »gleichzeitige« Mehrfach-Tastenabfrage ist genau genommen nicht ganz gleichzeitig.

Nichts in einem einzelnen Computer ist gleichzeitig! Alles erfolgt in einer Sequenz.

In dem Spielprogramm für vier Personen erfolgt die Tastenabfrage

**Listing 2. Ein Tastenspiel für vier Personen**

```
0 REM****************
1 REM** SPIEL FUER **
2 REM** 4 PERSONEN **
3 REM****************
4 REM
5 REM
6 REM****************
7 REM*** EINGABE ****
8 REM**MASCHINENCODE*
9 REM****************
10 PRINT CHR$(147)
20 POKE 51,235
30 POKE 52,29
40 POKE 55,235
50 POKE 56,29
VC 20: 100 DATA 169,0,141,32,145,173,33,145,141,255,29,96
C 64: 100 DATA 169,0,141,0,220,173,1,220,141,255,29,96
110 FOR K=1 TO 12
120 READ X
130 POKE 7660+K,X
140 NEXT K
199 REM**************
200 REM*ANWEISUNGEN**
205 REM**************
210 PRINT CHR$(147)
215 POKE830,0
VC 20:
220 PRINT"SPIELER 1:(W)TASTE
230 PRINT"SPIELER 2:(Z)TASTE
240 PRINT"SPIELER 3:(DEL)TASTE
250 PRINT"SPIELER 4:(CRSR←)TASTE
C 64:
220 PRINT"SPIELER 1 DRUECKT '←'
230 PRINT"SPIELER 2 DRUECKT 'COMMODORE'
240 PRINT"SPIELER 3 DRUECKT 'INST/DEL'
250 PRINT"SPIELER 4 DRUECKT 'CRSR←'
260 PRINT TAB(125)"SPACE"
270 GET A$:IF A$=""THEN 270
280 PRINT CHR$(147)
300 POKE 36879,8
310 PRINT"WER SIEHT ALS ERSTER
320 PRINT"DEN BUCHSTABEN
330 BU = INT(RND(0)*10)+1
VC 20:
340 POKE 7808,BU
350 POKE 38528,7
360 PRINT"UND DRUECKT SEINE   TASTE ?
C 64:
340 POKE 1224+17,BU
350 POKE 55496+17,7
360 PRINT"UND DRUECKT SEINE TASTE ?
370 PRINT TAB(165)"SPACE"
380 GET A$:IF A$="" THEN 380
390 PRINT CHR$(147)
400 FOR T=1 TO 800:NEXT T
```

Listing 1. In diesem Programm können alle Tasten abgefragt und angezeigt werden ►

```
4 REM **********************
5 REM   VIELFACHTASTEN
6 REM      ABFRAGE
7 REM **********************
10 PRINT CHR$(147)
20 POKE 51,235
30 POKE 52,29
40 POKE 55,235
50 POKE 56,29
100 DATA 169,254,141,0
101 DATA 220,173,1,220
102 DATA 141,255,29,96
105 REM --------VC20----------
106 REM   100 DATA169,254,141,32
107 REM   101 DATA145,173,33,145
108 REM   102 DATA141,255,29,96
109 REM --------VC20-ENDE------
110 FOR K=1 TO 12
120 READ X
130 POKE 7660+K,X
140 NEXT K
200 POKE 7662,127
210 SYS 7661
220 PRINT " ■"PEEK(7679);
300 POKE 7662,191
310 SYS 7661
320 PRINT PEEK(7679);
400 POKE 7662,223
410 SYS 7661
420 PRINT PEEK(7679);
500 POKE 7662,239
510 SYS 7661
511 REM  ------VC20-----------
512 REM  520 PRINT PEEK(7679)
513 REM  550 GET A$
514 REM  560 IF A$<>"■"THEN200
515 REM  ------VC20-ENDE--------
520 PRINT PEEK(7679);
600 POKE 7662,247
610 SYS 7661
620 PRINT PEEK(7679);
700 POKE 7662,251
710 SYS 7661
720 PRINT PEEK(7679);
800 POKE 7662,253
810 SYS 7661
820 PRINT PEEK(7679);
900 POKE 7662,254
910 SYS 7661
920 PRINT PEEK(7679)
940 REM -------VC20-------------
950 REM  GET A$
960 REM  IF A$<>"■" THEN 600
970 REM ------VC20-ENDE--------
1000 GOTO 200

READY.
```

```
490 REM**************
495 REM* BUCHSTABEN *
500 REM** WUERFELN **
510 REM**************
520 VB=INT(RND(0)*10)+1
530 IF VB=BU THEN POKE 830,250
VC 20:
540 C=INT(RND(0)*505)
550 POKE 7680+C,VB
560 POKE 38400+C,1
C 64:
540 C=INT(RND(1)*501)*2
550 POKE 1024+C,VB
560 POKE 55296+C,1
590 REM**************
595 REM** ABFRAGE ***
600 REM* DER TASTEN *
605 REM**************
610 FOR Z=1 TO 15
VC 20:
620 POKE 7662,253
630 SYS 7661
640 IF PEEK(7679)=253 AND PEEK(830)=250 THEN 810
650 POKE 7662,239
660 SYS 7661
670 IF PEEK(7679)=253 AND PEEK(830)=250 THEN 820
680 POKE 7662,254
690 SYS 7661
700 IF PEEK(7679)=127 AND PEEK(830)=250 THEN 830
710 POKE 7662,251
720 SYS 7661
730 IF PEEK(7679)=127 AND PEEK(830)=250 THEN 840
740 NEXT Z
C 64:
620 POKE 7662,127
630 SYS 7661
640 IF PEEK(7679)=253 AND PEEK(830)=250 THEN 810
650 IF PEEK(7679)=223 AND PEEK(830)=250 THEN 820
660 IF PEEK(7679)=221 AND PEEK(830)=250 THEN 850
670 POKE 7662,254
690 IF PEEK(7679)=254 AND PEEK(830)=250 THEN 830
700 IF PEEK(7679)=251 AND PEEK(830)=250 THEN 840
710 IF PEEK(7679)=250 AND PEEK(830)=250 THEN 850
720 NEXT Z
750 GOTO 500: REM (NEUER BUCHSTABE)
799 REM**************
800 REM** ERGEBNIS **
805 REM**************
810 PRINT"■■■■SPIELER 1 IST SIEGER":GOTO 860
820 PRINT"■■■■SPIELER 2 IST SIEGER":GOTO 860
830 PRINT"■■■■SPIELER 3 IST SIEGER":GOTO 860
840 PRINT"■■■■SPIELER 4 IST SIEGER":GOTO 860
C 64:
850 PRINT"■■■ZWEI SPIELER WAREN GLEICHZEITIG...
860 PRINT"■■■NOCH EINMAL ?  (J/N)
870 GET A$:IF A$="J" THEN 200
880 IF A$="N" THEN END
890 GOTO 870
READY.
```

▲ Ein Spiel für 4 Personen

natürlich auch hintereinander, wodurch die zuerst abgefragten Tasten ihren Besitzern einen kleinen Vorteil gewähren, aber halt nur einen ganz winzigen!

Dieser Effekt wird durch das 15fache Durchlaufen der Abfrage und durch die Schnelligkeit des Maschinenprogramms gemildert. Das ganze Programm in Maschinensprache geschrieben würde natürlich durch die Geschwindigkeit auch die letzte Ungerechtigkeit ausmerzen.

So, liebe VC 20er und C 64er. Das war's.

Ich hoffe, Sie kennen jetzt die verschiedenen Codes und fühlen sich wohl bei der Tasten-Abfrage. Methoden und Kochrezepte haben Sie dazu ja jetzt genug.

Ich habe mich bemüht, Sie zum Experimentieren anzuregen, denn gerade diese spielhafte Auswirkung der Neugier unterscheidet den Amateur vom Profi und fördert das Verständnis der Zusammenhänge.

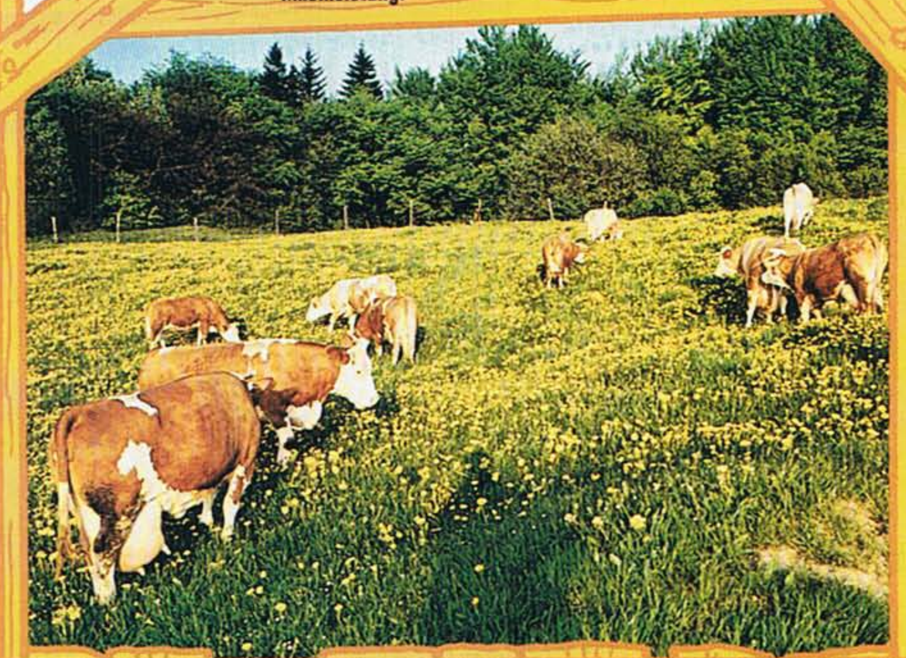
Ich habe mir am Anfang das Ziel gesetzt, allgemein verständlich und ohne Fachchinesisch zu schreiben.

Ob mir das gelungen ist, können nur Sie mir sagen. Falls etwas unklar geblieben ist und Sie Fragen haben, schicken Sie sie mir, ich werde Ihnen antworten. Und wenn es sogenannte »gute Fragen« sind, schreibe ich vielleicht darüber den nächsten Kurs.

(Dr. Helmuth Hauck/gk)

Literatur:
1. VIC REVEALED von N. Hampshire, Computabits Ltd., 1981
2. M. Bassman, S. Lederman in COMPUTE!S FIRST BOOK OF VIC COMPUTE!, Books Publ., 1982
3. VC 20 SPIELE-BUCH 1 von A. Dripke, Computer Life Verlag, 1983
4. VC-INTERN von Angerhausen und Englisch, Data Becker, 1983
5. 64-INTERN von Angerhausen et al., Data Becker, 1983
6. DAS VC-20-Buch von M. Hegenbarth, M. Schäfer, Markt & Technik Verlags, 1983
7. VIC-20-PROGRAMMERS REFERENCE GUIDE von A. Finkel et al., Howard W. Sams & Co, 1982

Gras allein macht Kühe nicht glücklich. Ohne das richtige Ergänzungsfutter sinkt die Milchleistung.



# Der Computer im Kuhstall

**Auch in der Landwirtschaft läßt sich ein Heimcomputer sinnvoll einsetzen.**

**Ein Agrar-Ingenieur aus Baden-Württemberg berechnet mit seinem C 64 die Futterration für das Milchvieh.**

Die Kunst der Fütterung von Milchkühen besteht nicht darin, ihnen etwas Freßbares in den Futtertrog zu werfen und dann zu warten, daß am Euter möglichst viel Milch herauskommt, meint Willi Lackenbauer. »Das Futter muß ausgewogen sein, sonst geben die Kühe nicht genug Milch«. Der Fachmann gibt seit zehn Jahren Landwirten Tips und berät sie bei der Viehhaltung.

Seit sechs Monaten setzt Willi Lackenbauer einen C 64 ein, um die Milchleistung in den Kuhställen zu optimieren. Für jede Kuh erstellt er einen maßgeschneiderten Plan. »Gras allein macht Kühe nicht glücklich — und nur glückliche Kühe geben viel Milch, das weiß sogar der Städter« meint Willi Lackenbauer. Er erläutert die Grundzüge der Fütterung: »Es kommt vor allem auf eine ausgewogene Nahrung an. Das ist beim Menschen ja nicht anders. Wer sich überwiegend von Fett und Kohlenhydraten ernährt, wird in seiner Leistungsfähigkeit nachlassen, weil ihm Eiweiß fehlt. In ähnlicher Weise wird auch das Futter für Kühe nach sogenannten Nährstoffeinheiten bewertet. Diese sind — nach Futtersorten geordnet — in Tabellen festgehalten. Bei der Rationsberechnung muß man davon ausgehen, daß die Kuh einen bestimmten Nährstoffanteil für ihren Grundbedarf benötigt. Erst was darüber hinaus gefüttert wird, wandelt sich in Leistung — sprich Milch — um. Dabei steht ziemlich genau fest, wieviel Nährstoffe für einen Liter Milch notwendig sind.«

Der Milchvieh-Experte gibt hierzu ein Beispiel: »Grünfutter ist ein ausgesprochener Eiweißlieferant, jedoch arm an Energie. Ohne Energie wird das pflanzliche Eiweiß nicht in tierisches Milcheiweiß umgewandelt oder Milchfett gebildet. Um das im Grundfutter enthaltene Eiweiß für die Milchleistung zu nutzen, muß energiereiches Kraftfutter zugefüttert werden.«

WIRTSCHAFTLICHE MILCHVIEHFUETTERUNG

RATIONSBERECHNUNG FUER MILCHKUEHE

LACKENBAUER WILLI
REBGUTSTR.38
6970 L A U D A

GRUNDFUTTERRATION JE KUH UND TAG:

| | | |
|---|---|---|
| ROTKLEE I.SCHN. | KG | 15.00 |
| WIESE I.SCHN.(IN BL.) | KG | 15.00 |
| MAISSILAGE 32%TS | KG | 12.00 |
| HAFERSTROH | KG | 1.00 |
| WIESENHEU I.SCHN.(I.BL.) | KG | 2.00 |

DIESE RATION ENTHAELT:

| | | |
|---|---|---|
| TROCKENSUBST | 12720.00 | GRAMM |
| ENERGIE | 73.40 | MJNEL |
| ROHPROTEIN | 1612.00 | GRAMM |
| ROHFASER | 3218.20 | GRAMM |
| CALCIUM | 106.40 | GRAMM |
| PHOSPHOR | 30.60 | GRAMM |

MILCHLEISTUNG AUS GRUNDFUTTER:

| | | |
|---|---|---|
| ENERGIE | 11.26 | KG MILCH |
| ROHPROTEIN | 13.31 | KG MILCH |
| CALCIUM | 25.12 | KG MILCH |
| PHOSPHOR | 2.70 | KG MILCH |
| CA:P-VERH. | 3.49 | :1 |

AUSGLEICHS-U.LEISTUNGSFUTTER:

| | | |
|---|---|---|
| WINTERGERSTE | KG | 2.00 |
| MILCHLEISTUNGSF.16%(III) | KG | 2.00 |

DIE GESAMTRATION ENTHAELT:

| | | |
|---|---|---|
| TROCKENSUBST | 16240.00 | GRAMM |
| ENERGIE | 101.64 | MJNEL |
| ROHPROTEIN | 2142.00 | GRAMM |
| ROHFASER | 3578.00 | GRAMM |
| CALCIUM | 127.60 | GRAMM |
| PHOSPHOR | 50.60 | GRAMM |

MILCHERZEUGUNGSWERT DER GESAMTRATION:

| | | |
|---|---|---|
| ENERGIE | 20.17 | KG MILCH |
| ROHPROTEIN | 19.55 | KG MILCH |
| CALCIUM | 31.75 | KG MILCH |
| PHOSPHOR | 14.47 | KG MILCH |
| CA:P-VERH. | 2.52 | :1 |

DIE RATION ENTHAELT 22.03 % ROHFASER IN DER TR.SUBST

**Dieser Ausdruck zeigt, welche Werte bei einer wirtschaftlichen Rationsberechnung wichtig sind.**

Mit der Kuhhaltung auf einem klassischen Bauernhof hat die moderne Milchviehhaltung nichts mehr zu tun. Seit Anfang der 60er Jahre hat sich beim Neubau von Ställen der sogenannte Boxenlaufstall durchgesetzt. Vierzig, achtzig und mehr Kühe in einem Stall. Jede Kuh kann frei herumlaufen. Will sie sich hinlegen, geht sie in ihre eigene Liegebox. Die Rangordnung in der Herde bestimmt, welcher Kuh welche Box gehört. Außerdem gibt es in einem Stall noch zwei ganz besondere Stände: den Melkstand und den Futterstand. Im Melkstand steht der Melker vertieft und hat die Euter in Griffhöhe vor sich. Ob mit der Melkmaschine oder — im Ausnahmefall — von Hand, die Kuh ist bequemer zu melken. Außerdem ist es recht praktisch, nicht er geht zur Kuh wie früher, sondern die Kuh kommt, um sich melken zu lassen. Damit sie ruhig steht, wird sie währenddessen mit Futter abgelenkt.

## Heimlich »naschen« ist unmöglich

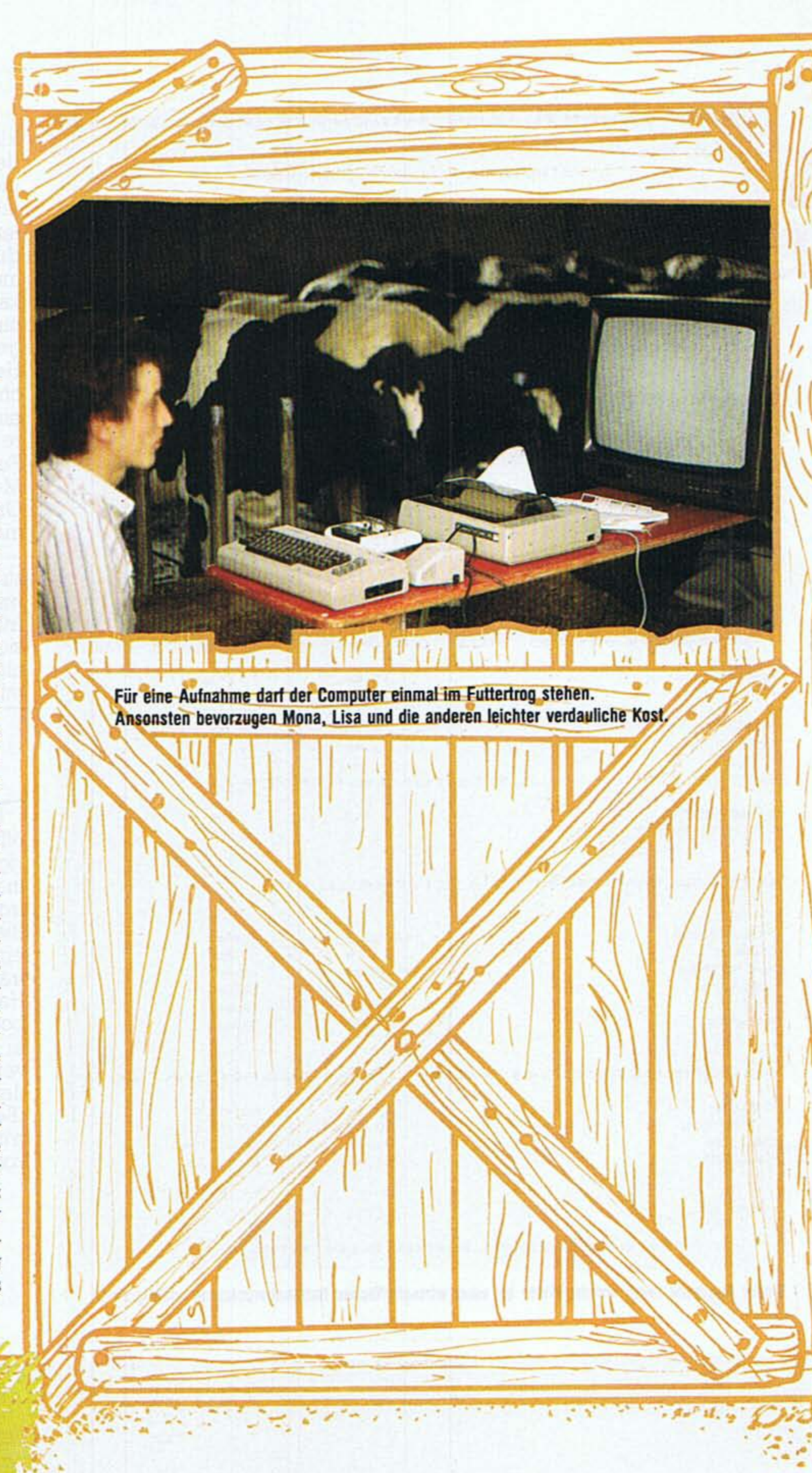
Der Futterstand ist ganztägig »geöffnet«. Immer wenn eine Kuh hungrig ist, geht sie fressen. Aber wie will man bei einer so großen Herde kontrollieren, wieviel jede frißt? »Da hat die Elektronikindustrie schon längst eine Lösung gefunden. Jede Kuh trägt einen elektronischen Code am Halsband, einen sogenannten Transponder«, erklärt Willi Lackenbauer schmunzelnd. Für ihn ist es selbstverständlich — doch der Laie wundert sich: Elektronik im Kuhstall. »Beim Eintreten in den Futterstand wird die Kuh identifiziert und bekommt eine Portion von 300 Gramm.



Wegen der Verdauung ist es günstiger, das Kraftfutter in möglichst viele Portionen aufzuteilen. Hat eine Kuh ihre Tagesration aufgebraucht, bekommt sie auch bei mehrfachem Besuch des Futterstandes nichts mehr. Was jede Kuh tatsächlich gefressen hat, kann sich der Landwirt jeden Abend auf einem Ausdruck anschauen. Die Soll-Menge für jede Kuh richtet sich nach ihrer Milchleistung und wird in wöchentlichem Turnus angepaßt.«

## Zwei Minuten statt zwanzig

Der Landwirt braucht sich nicht zu sorgen, solange seine Kühe soviel fressen wie sie sollen und die erwartete Menge Milch geben. Doch bei »Problemkühen« wird Willi Lackenbauer häufig um Rat gefragt. Er hilft herauszufinden, was bei Kühen nicht stimmt, die ihr Kraftfutter nicht aufgebraucht haben. »Ich brauche Angaben, was und wieviel die Kuh täglich frißt. Daraufhin berechne ich, von welchem Stoff sie zuviel oder zuwenig genommen hat. Eine Berechnung der Futterration dauerte im Normalfall etwa zwanzig Minuten. Das war mir irgendwann zu langweilig, denn ich brauche die Ergebnisse möglichst rasch für das eigentliche Beratungsgespräch.« Willi Lackenbauer griff auf die Elektronik zurück: Im Juli letzten Jahres kaufte er sich einen C 64 mit Kassettenrecorder. Für die Grundausstattung reichte ihm zunächst ein Fernseher zur Datenausgabe. Die erste Zeit war nicht besonders rosig: Probieren und Üben standen auf dem Programm — denn mit der bekannten Versuchs-und-Irrtums-Strategie kam er besser mit seinem Computer zurecht als durch den »Ratgeber« Handbuch. Im Januar erweiterte der inzwischen versierte Hobby-Programmierer seine Erstausstattung mit einem Drucker. Damit ging das



**Für eine Aufnahme darf der Computer einmal im Futtertrog stehen. Ansonsten bevorzugen Mona, Lisa und die anderen leichter verdauliche Kost.**

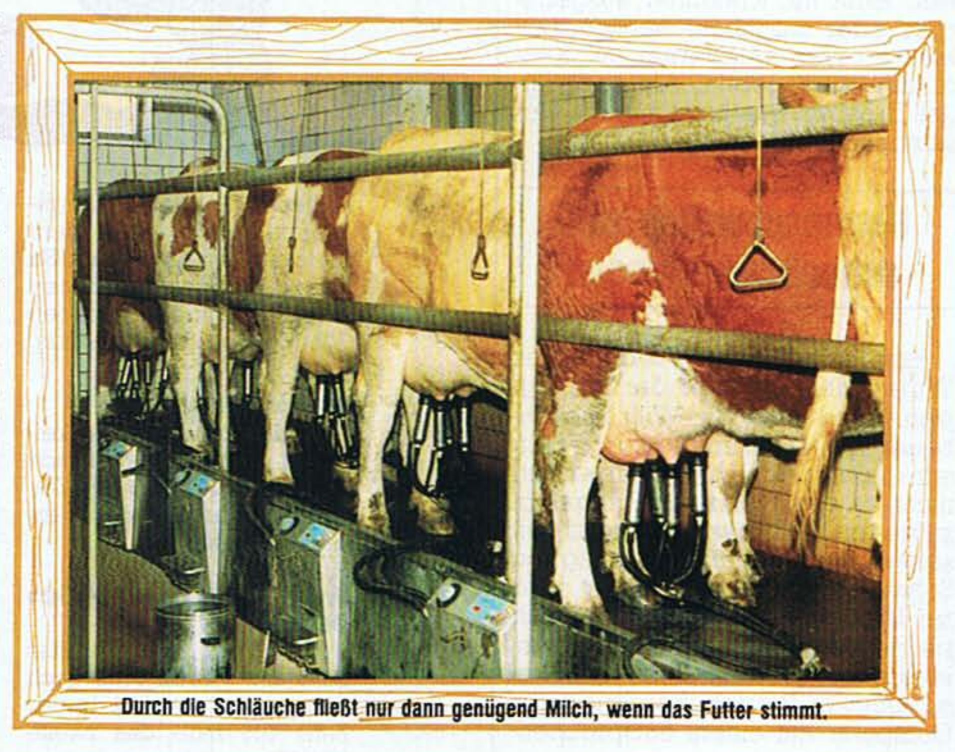
Überarbeiten der Listings wesentlich schneller als am Bildschirm. Kurze Zeit später war auch das erste Programm, die »Rationsberechnung für Milchkühe«, fertig.

Seither ist der C 64 ein unentbehrlicher Assistent für Willi Lackenbauer: »Umständliches Heraussuchen von Tabellenwerten kenne ich nicht mehr. Das Multiplizieren der Daten mit der Futtermenge und das Addieren von Spalten erledigt der Computer in Windeseile. Mit der Dateneingabe dauert eine Rations-Berechnung jetzt nur noch knapp zwei Minuten. Das ist ein Zehntel der Zeit, die ich früher brauchte.«

Aus den Ergebnissen kann Willi Lackenbauer wichtige Schlüsse ziehen: »Ein Eiweißüberschuß oder das Verhältnis der Mineralstoffe im Futter sind Hinweise auf Fruchtbarkeitsstörungen. Und eine unfruchtbare Kuh gibt auch keine Milch. So merkwürdig sich das anhört, diese Störungen lassen sich unter Umständen durch billigeres Energiefutter ausgleichen. Medikamente oder teures Kraftfutter sind oftmals gar nicht nötig«.

Neben seiner Berater-Tätigkeit kümmert sich Willi Lackenbauer auch um den Nachwuchs. Er ist Lehrer an einer landwirtschaftlichen Fachschule. Sein Programm setzt er natürlich auch im Unterricht ein. Bei Klassenarbeiten zum Thema Futterrationen müssen die Schüler zeigen, daß sie Futtermengen für Milchvieh selbst zusammenstellen und berechnen können. Heute kann der Lehrer jedem seiner Schüler individuelle »Problemkühe« auf dem Papier vorsetzen. Die Korrektur der Arbeiten ist mit dem C 64 ein Kinderspiel: Binnen zwei Minuten weiß er, ob die jeweilige Berechnung richtig ist. Vorher wäre es unmöglich gewesen, bei jedem Schüler andere Aufgaben zu überprüfen.

»So wie ich als Berater und Lehrer, kann jeder Milchviehhalter eine Menge Zeit sparen. Der Computer ist ohne Zweifel eine lohnenswerte Anschaffung. Die Futterrationen müssen nämlich nicht nur bei Problemkühen berechnet werden, sondern bei jedem Wechsel des Grundfutters. Das hängt von der Jahreszeit ab. Einmal ist es frisches Gras, dann Heu oder Futter aus dem Silo. Wer eine optimale Fütterung betreiben will, muß enorm viel Zeit für Rechnerei opfern. Bei nur fünfzig Kühen im Stall sind das schon 500 Stunden im Jahr. Mit dem Computer hat er — bei dreimaligem Wechsel des Grundfutters im Jahr eine Menge Zeit gespart.« Willi Lackenbauer ist überzeugt, daß jeder Landwirt nach kurzer Einarbeitung mit dem Programm umgehen kann. Er hat auch schon die ersten Belege dafür. Zwei Betriebe setzen sein Programm seit etwa einem halben Jahr ein und haben seither die Fütterung fest im Griff. Futtervergeudung gehört dort der Vergangenheit an.



Durch die Schläuche fließt nur dann genügend Milch, wenn das Futter stimmt.

Das Programm »Rationsberechnung für Milchkühe« ist in Basic für den C 64 geschrieben. Es benötigt etwa 15 KByte Speicherkapazität. Auf PEEKS und POKES wurde verzichtet, um eine Übertragung auf den kleineren Bruder des C 64, den VC 20 zu erleichtern. Als Speichermedium wurde — aus Kostengründen — ein Kassettenrecorder gewählt. Wer auf Ausdrucke keinen Wert legt, bleibt bei den Hardwarekosten unter 1000 Mark. Das Programmlisting kann bei der Redaktion angefordert werden.

## Grundausstattung für weniger als 1000 Mark

Die Bedienung ist wirklich kinderleicht. Nach dem Erscheinen des Titels werden zunächst Name und Adresse abgefragt. Diese Eingabe ist kein Muß — sie kann auch übergangen werden. Dann erscheinen drei mal fünfzehn Grundfuttermittel

## So machen's andere

auf dem Bildschirm in folgender Reihenfolge: Grünfutter, Silagen (Futter aus dem Silo), Heu und Stroh. Hier muß die Menge des ausgewählten Futtermittels in Kilogramm eingegeben werden. Fehleingaben können korrigiert werden. Zum Abschluß erscheint die eingegebene Tagesration für jede Kuh. Mit »N« kann man an den Anfang zurückgehen. Sind alle Eingaben bestätigt, erscheinen nach kurzer Rechendauer die Inhaltsstoffe und der Milcherzeugungswert der Ration. Nun kann mit Kraftfutter ausgeglichen oder aufgebaut werden. Die Menge wird wiederum in Kilogramm eingegeben. Als endgültiges Ergebnis erscheint nun die Gesamtration.

### Hemmschwellen sind schnell überwunden

Rationalisierung und vernünftiges Wirtschaften ist einer der wichtigsten Grundsätze, die Willi Lackenbauer in Theorie und Praxis vermitteln muß und will. Und so gab er sich nicht mit einem einzigen Programm zufrieden. Mittlerweile kann er mit seinem C 64 auch die Fütterung von Schweinen, egal ob zur Zucht oder zur Mast im Stall, optimieren. Auch bei der Berechnung von Deckungskosten und Gewinn eines landwirtschaftlichen Betriebes assistiert der Computer mit einem entsprechenden Programm.

Willi Lackenbauer glaubt, daß Computer in der Landwirtschaft in Kürze einen festen Platz haben werden: »In wenigen Jahren wird der Transponder in vielen Boxenlaufställen zum Standard gehören. Junge Landwirte bekommen bereits in ihrer Ausbildung den ersten Kontakt zu Computern. Hemmschwellen werden dadurch schnell abgebaut. Fütterungsprogramme, die auf einem preiswerten Heimcomputer wie dem C 64 laufen, werden sich mit Sicherheit durchsetzen. Für den Berater ist es auch besser, seine Zeit effektiver als für schematische Berechnungen zu nutzen.« Der computerbegeisterte Berater und Lehrer plädiert für Programme, die einfach und nachvollziehbar landwirtschaftlichen Betrieben Routinearbeiten abnehmen. »Die eingesparte Zeit kann viel sinnvoller zur Beobachtung und Betreuung der Tiere genutzt werden, denn das kann kein Computer dem Landwirt jemals abnehmen.« (Willi Lackenbauer/kg)



## Fehlerteufelchen

Dieses Mal ist es mir gelungen, mich mit dem Drucker zu verbünden! So konnte ich Ihnen auch in den Ausgaben 6 und 7 das Leben (Eintippen) schwermachen.

**Listing Quicktext 6/84, Seite 60**

Da der Epsondrucker keinen Pfeil links und Pfeil nach oben versteht, muß das Zeichen »_« durch Pfeil links ersetzt werden.

**Listing Kurvendiskussion 7/84, Seite 116**

In diesem Listing muß das Zeichen ^ durch Pfeil nach oben ersetzt werden. In der Zeile 316 muß das zweite PRINT um #4 ergänzt werden. Die Zeile 266 fehlt nicht.

**Listing Hardcopy 1520 7/84, Seite 108**

Auch hier muß das Zeichen ^ durch Pfeil nach oben (Potenz) ersetzt werden.

**Listing Softwarekatalog 7/84, Seite 72**

**Listing Q-Bernd 7/84, Seite 142**

In diesen Listings muß der reverse Querstrich durch das reverse Pfund-Zeichen (Farbe Rot) ersetzt werden.

**Testbericht »Daten gut im Griff« Ausgabe 5/84, Seite 52**

Datamat: Entgegen früheren Angaben wird die neue Datamatversion nicht für zirka 25 Mark umgetauscht. Der Umtauschpreis beträgt jetzt 50 Mark.

Multidata: Zwei Tage nach dem das 64'er-Magazin mit diesem Testbericht erschienen war, mußte ich feststellen, daß meine Angaben nicht ganz stimmten. Multidata ist doch nicht so absturzsicher wie ich geschrieben habe. Man kann bei bestimmten Abfragen das Programm durch falsche Eingaben zum Absturz bringen.

Weiterhin sollte man es vermeiden, sich einen anderen Monitor als den von Commodore anzuschaffen, denn durch die etwas merkwürdige Farbauswahl ist die Schrift sonst kaum noch zu lesen.

**Listing »Terminalprogramm« 7/84, Seite 24**

Leider sind bei diesem Programm die beiden »Wandler-Programme« nicht mit abgedruckt worden. Hier nun die Ergänzung.

Programm **"WANDLER"**:

```
60000 PRINT"(rev.Herz)(rev.Q)": SYS16964:PRINT"RUN 60020"
60010 PRINT"(rev.S)":END
60020 PRINT"(rev.Herz)(rev.Q 2x)":SYS16970:
PRINT"RUN60020"
60030 PRINT"(rev.S)"
```

Programm **"WANDLER 2"**:

```
60000 PRINT"(rev.Herz)(rev.Q 2x)":SYS16964:
PRINT"RUN60020"
60010 PRINT"(rev.Herz)":
POKE631,13:
POKE632,13
60015 POKE198,2:END
60020 PRINT"(rev.Herz)(rev.Q 2x)":SYS16964:PRINT
"RUN 60020 "
60025 POKE198,2:END
60030 PRINT"(rev.Herz)"
```

Fortsetzung von Seite 54

wenn ein entsprechendes Ladeprogramm oder das Pascal selbst (G-Pascal) vorhanden ist. Betrachtet man die reine Compilierzeit, so schneidet das G-Pascal bei weitem am besten ab.

## KMMM-Pascal schnell und mit gutem Editor

Bezüglich der Ablaufgeschwindigkeit ist die Sache klar: Das KMMM-Pascal und das neue Pascal 64 V3.0 sind klar die schnellsten (über das Oxford Pascal ist hierzu noch nichts bekannt). Ich benutze das KMMM-Pascal, denn für den engagierten Pascal-Anwender gibt es außerdem immer noch unbequemen Pascal 64 V3.0 auf dem Commdore 64 bis heute noch nichts besseres, obwohl wahrscheinlich das Oxford Pascal in näherer Zukunft eine Leaderstellung erreichen wird, außer man will sich die CP/M-Karte kaufen. Hat man einmal das Problem mit dem Diskettenformat gelöst, dann ist allerdings die Frage nach einem geeigneten Pascal akademisch. Das KMMM-Pascal kostet in der Schweiz zirka 280 SFr., das Oxford Pascal zirka 190 SFr., was im Vergleich zu den anderen Versionen, bis auf das neue Pascal 64 V3.0, dessen Preis auch 99 Mark sein wird, etwas hoch erscheinen mag. Falls diese Preise dem Einsteiger zu hoch sind, dann kann ich ihm höchstens noch das G-Pascal empfehlen, weil es dem Anfänger wohl eher auf leichte Editierbarkeit ankommt, als auf ein möglichst komplettes Pascal (Pascal 64 V3.0), das ihm die übrigen Versionen sowieso auch nicht bieten können. Ein Pascalsystem sollte eben nicht nur aus einem Compiler (Data Becker Pascals) bestehen, sondern auch einen guten Editor (KMMM-Pascal, G-Pascal, Oxford-Pascal) besitzen, womit man frustrationslos und einfach Sourcefiles erstellen kann. Pascal ist eben (leider!!) eine Compilersprache.

(Martin Baur)

## Inserentenverzeichnis



## Impressum


**Herausgeber:** Carl-Franz von Quadt, Otmar Weber

**Chefredakteur:** Michael M. Pauly (py)
**Stellv. Chefredakteur:** Michael Scharfenberger (sc)
**Redakteure:** aa = Albert Absmeier, leitender Redakteur (130), ev = Volker Everts (278), kg = Karin Gößlinghoff (269), gk = Georg Klinge (130), rg = Christian Rogge (278)
**Redaktionsassistenz:** Dagmar Zednik (237)
**Fotografie:** Janos Feister, Titelfoto: Alex Kempkens

**Layout:** Leo Eder (Ltg.), Willi Gründl, Walter Höß, Cornelia Weber

**Auslandsrepräsentation:**
**Schweiz:** Markt&Technik Vertriebs AG, Alpenstrasse 14, CH-6300 Zug, Tel. 042-223155/56, Telex: 862329 mut ch
**USA:** M&T Publishing, 2464 Embarcadero Way, Palo Alto, CA 94303; Tel. 001-4240600; Telex 752351

**Manuskripteinsendungen:** Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten werden, so muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt & Technik Verlags AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

**Herstellung:** Klaus Buck (180), Leo Eder (181)

**Anzeigenleitung:** Peter Schrödel (156)
**Anzeigenverkauf:** Alfred Reeb (211)

**Anzeigenverwaltung und Disposition:** Michaela Hörl (171)

**Anzeigenformate:** 1/1-Seite ist 266 Millimeter hoch und 185 Millimeter breit (3 Spalten à 58 mm oder 4 Spalten à 43 Millimeter). Vollformat 297 x 210 Millimeter. Beilagen und Beihefter siehe Anzeigenpreisliste.

**Anzeigenpreise:** Es gilt die Anzeigenpreisliste Nr. 1 vom 1. März 1984.
**Anzeigengrundpreise:** 1/1 Seite sw: DM 7400,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1000,-. Vierfarbzuschlag DM 3000,-. Plazierung innerhalb der redaktionellen Beiträge: Mindestgröße 1/2-Seite
**Anzeigen im Einkaufs-Magazin:** Die ermäßigten Preise im Einkaufs-Magazin gelten nur innerhalb des geschlossenen Anzeigenteils, der ohne redaktionelle Beiträge ist. 1/1-Seite sw: DM 5400,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1000,-. Vierfarbzuschlag DM 3000,-. **Anzeigen in der Fundgrube: Private Kleinanzeigen** mit maximal 5 Zeilen Text DM 5,- je Anzeige. **Gewerbliche Kleinanzeigen:** DM 10,- je Zeile Text.
Auf alle Anzeigenpreise wird die gesetzliche MwSt. jeweils zugerechnet.

**Vertriebsleitung, Werbung:** Hans Hörl (114)

**Vertrieb Handelsauflage:** Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebs GmbH, Plieninger Straße 100, 7000 Stuttgart 80 (Möhringen), Telefon (07 11) 72004-0

**Erscheinungsweise:** 64'er, Magazin für Computerfans, erscheint monatlich, Mitte des Vormonats.

**Bezugsmöglichkeiten:** Leser Service: Telefon 089/46 13-1 19. Bestellungen nimmt der Verlag oder jede Buchhandlung entgegen. Das Abonnement verlängert sich zu den dann jeweils gültigen Bedingungen um ein Jahr, wenn es nicht zwei Monate vor Ablauf schriftlich gekündigt wird.

**Bezugspreise:** Das Einzelheft kostet DM 6,-. Der Abonnementspreis beträgt im Inland DM 72,- pro Jahr für 12 Ausgaben. Darin enthalten sind die gesetzliche Mehrwertsteuer und die Zustellgebühren. Der Abonnementspreis erhöht sich um DM 18,- für die Zustellung im Ausland, für die Luftpostzustellung in Ländergruppe 1 (z.B. USA) um DM 38,-, in Ländergruppe 2 (z.B. Hongkong) um DM 58,-, in Ländergruppe 3 (z.B. Australien) um DM 68,-.

**Druck:** Druckerei E. Schwend GmbH, Schmollerstr. 31, 7170 Schwäbisch Hall





**Verantwortlich:** Für redaktionellen Teil: Michael M. Pauly.
Für Anzeigen: Peter Schrödel.

**Vorstand:** Carl-Franz von Quadt, Otmar Weber

**Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:**
Markt&Technik Verlag Aktiengesellschaft, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon 089/46 13-0, Telex 522052

Mitteilung gem. Bayerischem Pressegesetz: Aktionäre, die mehr als 25% des Kapitals halten: Otmar Weber, Ingenieur, München; Carl-Franz von Quadt, Betriebswirt, München. Aufsichtsrat: Dr. Robert Dissmann (Vorsitzender), Karl-Heinz Fanselow, Hans-Jochen Wolf, Eduard Heilmayr.

**Telefon-Durchwahl im Verlag:**

**Wählen Sie direkt: Per Durchwahl erreichen Sie alle Abteilungen direkt. Sie wählen 089-46 13 und dann die Nummer, die in Klammern hinter dem jeweiligen Namen angegeben ist.**

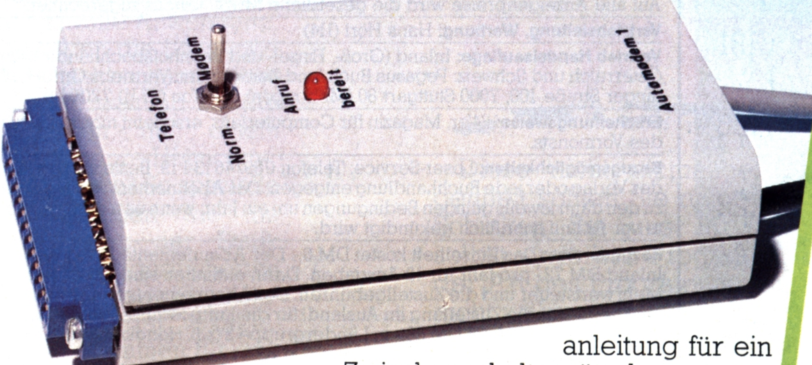
# VORSCHAU

## Die Musik macht der C 64

Eine herausragende Eigenschaft des Commodore 64 sind seine Musikfähigkeiten. Es gibt mittlerweile eine Vielzahl von Musikprogrammen und Keyboards, die den Synthesizer im C 64 ausnützen. Wir stellen die wichtigsten Musikprogramme, ihre Leistungen und Schwächen und einige Anwendungen vor.




Wie kann man einen Akustikkoppler zu einem Modem machen? Wir liefern Ihnen eine Bau-




anleitung für ein Zwischenschaltgerät, das automatisches Abheben und einen Wählautomaten beinhaltet.

## Software-Test

Diesmal haben wir uns die zwei besten Tabellenkalkulationsprogramme für den C 64 vorgenommen. Was bringen Calc Result und Multiplan auf einem 40-Zeichen-Bildschirm? Das Textverarbeitungsprogramm Textomat wurde erheblich verbessert und kostet immer nur noch 99 Mark. Wie steht Textomat nun im Vergleich zu wesentlich teureren Programmen da?

## Das schnellste Kopierprogramm für den C 64

Das hat es noch nicht gegeben: Durch Quickcopy wird eine voll gefüllte Diskette inklusive Formatieren in 2.5 Minuten (2 Laufwerke) oder 3.5 Minuten (1 Laufwerk) kopiert. Bei nicht ganz gefüllter Diskette geht es entsprechend schneller.

## Zwei neue Kurse

Endlich ist es soweit ! In der nächsten Ausgabe starten wir unseren großen Assembler-Kurs für C 64 und VC 20. Steigen Sie ein in die Welt der Maschinensprache-Programmierung — auch wenn Sie bisher ausschließlich in Basic programmiert haben: Dieser Kurs ist gerade für den Anfänger konzipiert worden.

Außerdem beginnen wir mit dem Abdruck der mehrteiligen Serie »Der gläserne VC 20«. Endlich wird der VC 20 auch für den Einsteiger durchschaubar gemacht. Ob Betriebssystem, Basic oder Grafik — dieser Kurs macht den VC 20 wirklich durchsichtig.

## Anwendung des Monats

Passend zu unserem Musikschwerpunkt ein Programm das Ihren Commodore 64 zur elektronischen Orgel werden läßt.

## Listing des Monats

»Spring-Vogel« ist ein Spielegenerator, mit dem sich beliebig viele Spielszenen selbst erstellen lassen.

## Listings

— Diskverwalter für C 64
— Ein deutscher Zeichensatz für den VC 20
— Spriteaid und Screen Change für den C 64
— Ein Schiebespiel für den VC 20
— List-Stop
— RS232-Test
— und wieder viele Tips & Tricks für VC 20 und C 64